

IES Palas Atenea

Proyecto de Investigación Bachillerato de excelencia:

Programación, redes y software libre

Invproy α

Un simulador de redes por y para alumnos

Adaptación para el II Encuentro Preuniversitario Jóvenes Investigadores

David Davó Laviña

Tutor

Julio Sánchez Olías

4 de enero de 2017


Índice general

Introducción	IV
1 Programación y software libre	1
1.1 Software Libre	1
1.2 Fundamentos básicos de programación en Python	3
2 Redes Informáticas	5
2.1 Capas de Red/Modelo OSI	6
2.2 Topologías de red	7
2.2.1 Clasificación de las topologías de red	7
2.2.2 Nodos de una red	8
2.3 Paquetes de red	10
2.4 Protocolos	11
2.4.1 Familia de protocolos de internet	11
2.5 Seguridad de redes	14
2.5.1 Tipos de ataques	14
2.5.2 Contramedidas	16
3 El simulador de redes	17
3.1 Herramientas usadas en la creación del software	17
3.1.1 GNU/Linux	17
3.1.2 Git y Github	18
3.1.3 LaTeX	20
3.1.4 Python	20
3.1.5 Gtk+	20
3.1.6 Wireshark	21
3.2 Instalación	21
3.2.1 Ubuntu / Debian	21
3.2.2 Arch Linux	21

3.2.3	Ejecución manual / instalación portable	22
3.3	Uso del programa	22
3.3.1	Configuración	24
3.3.2	Ejemplo: Envío de Ping entre dos dispositivos	25
3.4	Funcionamiento del programa	26
3.4.1	Main.py	26
3.4.2	save.py	31
3.4.3	Interface.glade	31
3.4.4	Dispositivos	32
3.5	Versión actual del programa (0.2.3-alpha)	34
3.6	Desarrollo del proyecto	34
3.6.1	Obstáculos en el desarrollo del proyecto	35
3.7	Conclusión	36
Bibliografía		37
Glosario y acrónimos		38
A Unidades de transferencia de datos		40
B Capturas de pantalla del programa		41
C Licencia GNU GPL		43
D Código del programa		53
D.1	Main.py	53
D.2	Modules/logmod.py	85
D.3	Modules/save.py	86
D.4	Config.ini	88

Índice de figuras

2.1	Representación esquemática de las diferentes topologías de red.	8
2.2	Encapsulación de red. El Datagrama IP es lo considerado 'Paquete de red'	10
2.3	Captura de pantalla de Wireshark	11
3.1	<i>Gitflow</i> o flujo de trabajo de Git	18
3.2	Interfaz de InvProy Alpha	23
3.3	Menú de Información de Dispositivos	24
3.4	Diagrama de flujo de la función <i>compcon</i>	29
B.1	Captura: Click derecho en un computador	41
B.2	Captura: Ventana para enviar ping.	41
B.3	Captura: Igual que B.2, pero con una IP válida.	41
B.4	Captura: Ventana con la tabla que posee el Switch.	41
B.5	Captura: Ventana de edición de propiedades de objeto.	41
B.6	Captura: Paquetes viajando por una red de ejemplo.	42

Todas las imágenes son de autoría propia y siguen la licencia de este documento. CC BY-SA 4.0 

Introducción

En el mundo contemporáneo, ninguna de las innovaciones tecnológicas sería posible sin algo fundamental: las redes; y, más concretamente, redes informáticas. Las redes informáticas han hecho posible, desde su nacimiento, la comunicación de grandes sumas de datos a velocidades casi instantáneas entre sitios distantes. Al principio esta tecnología era usada entre universidades, acelerando el proceso de investigación al coordinarse unas universidades con otras mucho más rápidamente.

Más tarde, se extendió el uso de esta tecnología del uso militar y científico a todas las empresas y hogares, comenzando así una revolución tecnológica que aún no se ha conseguido parar. Con acceso instantáneo a cultura, entretenimiento, conocimiento, información y más de dos mil exabytes de ancho de banda¹ viajando por la red, se ha convertido en una herramienta de uso por la humanidad imprescindible para cualquier actividad.

La tecnología del mundo contemporáneo no habría sido posible, en parte, también gracias al software libre y al desarrollo colaborativo, pues ha permitido el desarrollo de sistemas operativos como GNU/Linux de la *Free Software Foundation* (Usado actualmente por el 90 % de los servidores de red) o CUPS, el software para servidores de administración de impresoras más completo y competente usado en la mayoría de oficinas.

Son dos cosas muy importantes, que apenas son enseñadas en las clases de la ESO y Bachillerato, por eso he creado InvProy, un pequeño simulador de redes con la ambición de enseñar tanto de redes como de programación. Podrán experimentar, de una forma sencilla y muy visual como funciona una red y cómo se comportan los distintos protocolos. También, al ser software libre, los alumnos podrán aprender sobre programación al observar el código y tener la licencia para modificarlo y colaborar en el desarrollo del programa. Aunque el programa este aún en fase Alpha (fase de desarrollo), ya tiene la base para que sea muy sencillo añadir más protocolos, funcionalidades o dispositivos de red. A día de hoy tiene como dispositivos los ordenadores, conmutadores y concentradores. En cuanto a paquetes de red, permite enviar un Ping, usando los protocolos ICMP, IPv4 y Ethernet.

¹Datos a Mayo de 2015. Fuente: Cisco-[1]

Capítulo 1

Programación y software libre

Propuesta

El objetivo de este proyecto es el desarrollo abierto y colaborativo a largo plazo de un software programado en Python de código libre con el que los alumnos puedan aprender tanto sobre redes como de programación. Debe soportar los protocolos más utilizados en la actualidad y permitir una gran personalización por los usuarios. Además debe ser compatible con los sistemas operativos Ubuntu, MaX y Windows, y ser de fácil instalación para el alumnado. Debe ser intuitivo y fácil de usar e incluir una gran documentación.

1.1 Software Libre

Según la Free Software Foundation “«Software libre» es el software que respeta la libertad de los usuarios y la comunidad. A grandes rasgos, significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. Es decir, el «software libre» es una cuestión de libertad, no de precio. Para entender el concepto, piense en «libre» como en «libre expresión», no como en «barra libre». En inglés a veces decimos «libre software», en lugar de «free software», para mostrar que no queremos decir que es gratuito.” – R. Stallman [2]

La idea de Software Libre nace con Richard Stallman en 1983, cuando anuncia la creación del Proyecto GNU (Sistema Operativo libre alternativo a Unix y BSD). En 1985 se publica el Manifiesto GNU en el que se declara la filosofía GNU, la definición de software libre y algunas ideas sobre copyleft, más tarde ese año se crea la Fundación del Software Libre (FSF por sus siglas en inglés). A su sistema operativo aún le faltaba una pieza bastante grande, a lo que en 1991 Linus Torvalds lanza el Kernel Linux, que licenció con la licencia GNU General Public License (GPL)[Ver anexo C]. A partir de aquí comenzaron a salir nuevas licencias, como la licencia Apache, o la del MIT. Algunos ejemplos de software libre son GNU/Linux, emacs, LaTeX, GIMP, GNOME, o los servidores Apache y las librerías MySQL, usadas en todo el mundo.

No debemos confundir ‘Software Libre’ con ‘Código abierto’, ya que, aunque el código pueda ser leído por todo el mundo no significa que el resto de personas tengan licencia para redistribuir y/o editar el código. Software libre es el que cumple las cuatro libertades del software libre. Según Richard Stallman las cuatro libertades son estas: [3], [4]

- **Libertad 0:** La libertad de ejecutar el programa cuando quieras, para cualquier propósito.
- **Libertad 1:** La libertad de estudiar cómo el programa funciona, y la posibilidad de cambiarlo para que se ejecute como tú desees. (Acceso al código del programa).
- **Libertad 2:** La libertad de redistribuir las copias para ayudar a tus colegas.
- **Libertad 3:** La libertad de distribuir copias de tu versión modificada a otras personas.

Una de las grandes ventajas del software libre, aparece en la educación. Es muy útil para aprender ya que, si un alumno tiene curiosidad sobre el programa que está usando, puede consultar el código fuente en internet. Además, al ser licencias gratuitas, se puede destinar ese presupuesto a otras áreas como el hardware o el profesorado. También es útil en el desarrollo, pues cualquier programador puede solucionar un error que afecta a todos los usuarios.

1.2 Fundamentos básicos de programación en Python

Una variable no es más que un hueco en la memoria del ordenador, reservado para algo que queremos recordar. Así, podemos establecer una variable llamada 'pi' con valor 3,1415..., para que luego, en una función, en lugar de escribir '3.14...' múltiples veces, sólo haga falta escribir 'pi'. Esto es un ejemplo bastante básico, ya que las variables pueden ser también valores booleanos¹, listas, diccionarios (listas de tipo clave:valor), objetos, series de caracteres... Las variables se establecen así: `variable = valor`.

Una función, es un conjunto de instrucciones, que, dados unos argumentos (o ninguno) realiza una serie de acciones y/o retorna información (un número, objeto, valor, etc.). Por ejemplo, podríamos crear una función `sum(a,b)`, que dados dos números, retorne la suma de estos. Se dice 'llamar a una función' cuando se le da al programa la instrucción de que ejecute la función. Para ello, usamos `funcion(argumento1, argumento2...)`. Los argumentos pueden ser cualquier cosa, desde una variable binaria (1 o 0, Verdadero o Falso) a un Objeto. Además, al terminar una función retorna un valor; que puede ser usado si la función es llamada para establecer una variable. (P.ej: `sumatorio = sum(3, 8)`)

Las clases sirven para crear objetos, por ejemplo, un Switch, un cable, una ventana, o una dirección MAC son objetos. Las clases contienen variables y funciones. Las variables pueden ser distintas para cada objeto, mientras que las funciones son las mismas. Así todos los conmutadores tienen una función que actúa cuando reciben un paquete, pero cada uno posee una dirección MAC distinta, unas coordenadas distintas, un nombre distinto... Para crear un objeto, simplemente tenemos que 'llamar' a la clase mediante una variable, p.ej: `objeto = Clase()`. Como la función que crea el objeto también acepta argumentos, hay objetos que necesitan argumentos para ser creados (`P = punto(5,7)`). También, podemos establecer variables de objeto: `objeto.nombre = "Objeto Guay"`.

Al final, todo programa se basa en condiciones y funciones. "Si ocurre esto, haz esto otro; si no ocurre, haz aquello". Para ello, se simplifica cualquier expresión en VERDADERO o FALSO, parecido a la lógica aristotélica. Si la expresión dada es verdadera, se ejecutará el código de dentro del condicional. También existe la expresión `elif`, donde se especifica otra condición, que se ejecutará en caso de que no se cumpla la anterior pero sí esta; y la expresión `else`, en caso de que no se cumpla ninguna de las anteriores.

¹Valores del álgebra de Boole; Verdadero/Falso, 1/0, True/False...


```
1 if condicion:
2     funcion()
3 elif condicion2:
4     funcion2()
5 elif condicionN:
6     funcionN()
7 else:
8     print("Algo no funciona")
```

Como condición, puede haber cualquier expresión que pueda ser verdadera. Las que unen condiciones son `and` y `or`, en lenguaje matemático, intersección y unión. Y las condiciones suelen ser expresiones matemáticas, como $a > b$, $a \geq b$, $a == b$, o $a != b$. Aunque en ocasiones pueden usarse con otros tipos de variables, como las series de caracteres. Por ejemplo `if entrada == "Yes":`, en la que si la variable `entrada`, que hemos definido anteriormente como texto introducido por el usuario, es igual a "Yes", ejecutaremos código. En mi programa tiene aplicaciones como 'Si el objeto es un conmutador' o 'si la coordenada x buscada es igual a la coordenada x del objeto y la coordenada y buscada es igual a la coordenada y del objeto'.

Ahora, veremos un pequeño ejemplo práctico, que junta todo:

```
1 def esnatural(numero): #Definimos la función, asumiendo que no hay números complejos
2     absoluto = abs(numero) #Absoluto es igual al resultado de la función absoluto del
    ↪ numero
3     entero = int(numero)
4     if absoluto == entero: #Si el número absoluto es igual al entero.
5         print("Es un número Natural") #Ponemos eso en la consola
6         return True
7     elif numero == entero: #No es positivo, por lo que miramos a ver si es negativo,
    ↪ pero entero.
8         print("Es un numero Entero, pero no Natural")
9         return False
10    else:
11        print("Es un numero Racional")
12        return False
```

Capítulo 2

Redes Informáticas

Historia

El uso de redes informáticas nace en la década de 1960, para suplir la necesidad de las universidades y laboratorios de investigación de conectar los distintos ordenadores. En la década de 1970 se comienza a experimentar con tecnologías de redes LAN, algunas de ellas usadas actualmente o recientemente, como Ethernet, desarrollado en 1975 por Xerox PARC (Palo Alto Research & Development).

Las redes se usaban sobre todo para aprovechar el almacenamiento y las impresoras, así no era necesario comprar una impresora para cada equipo. Cada vendedor incluía su propio tipo de tarjeta de red, cableado, protocolo y sistema operativo de red, hasta que Novell NetWare (Sistema Operativo de red desarrollado por Novell inc.) salió al mercado en 1983 soportando la mayoría de tipos de tarjetas de red y cables. Fue el SO de Red dominante hasta que en 1993 Microsoft lanzó Windows NT AS y Microsoft Windows para Trabajo en Grupo. Al mismo tiempo, los dispositivos Unix usaban sistemas basados en TCP/IP.

Actualmente el protocolo TCP/IP es considerado un estándar y ha reemplazado el resto de protocolos usados hasta principios de los 2000.

2.1 Capas de Red/Modelo OSI

El modelo OSI es un modelo de referencia para redes basado en capas de abstracción. Su objetivo es conseguir la interoperabilidad entre sistemas haciendo uso de los protocolos estandarizados. Fue creado en 1980 por la Organización Internacional de Estandarización (ISO). No es considerado una arquitectura de red porque los protocolos no forman parte del modelo, sino que son entidades de distintas normativas internacionales.

Capa	PDU ¹	Función	Ejemplos
1. Física	Bit	Transmisión y recepción de bits físicos sobre un medio físico (topología de red)	RJ45, IEEE 802.11, etc.
2. Data Link	Frame	Transmisión segura de <i>frames</i> entre dos nodos conectados por una capa física.	Ethernet, 802.11, etc...
3. Red	Paquete	Estructurar y administrar una red multinodo. Incluye enrutamiento, control de tráfico, y asignación de direcciones	IPv4, IPv6, ICMP...
4. Transporte	Datagrama(UDP) Segmento(TCP)	Transmisión de segmentos de datos entre los puntos de una red, incluyendo ACK	TCP, UDP...
5. Sesión	Datos	Administración de sesiones de comunicación, como intercambio continuo de información entre dos nodos.	SSH, RPC, PAP...
6. Presentación	Datos	Translación de datos entre un servicio de red y una aplicación. Incluye comprensión, encriptación/decriptación, y codificación de caracteres.	MIME, TLS
7. Aplicación	Datos	APIs de alto nivel, incluyendo recursos compartidos y acceso remoto de archivos	HTTP, FTP, SMTP...

¹Protocol Data Unit o Unidad de Datos de Protocolo.

2.2 Topologías de red

La topología de red es la configuración de los elementos que componen una red. Puede ser representada lógica o físicamente.

La topología lógica puede ser igual en dos redes, aunque su topología física (distancia entre conexiones, tipo de señales...) sea distinta. Se distinguen dos elementos: los nodos (distintos dispositivos de red) y los enlaces (medio de transmisión de los datos).

2.2.1 Clasificación de las topologías de red

Se distinguen ocho tipos de topologías de red lógicas: [5]

Punto a punto: conexión directa entre los dos puntos de la red. También es conocida como *P2P* (*Peer to Peer*).

Estrella: cada nodo está conectado a un nodo central que puede ser un enrutador, concentrador o conmutador.

Bus: cada nodo está conectado a un único cable. Una señal de un dispositivo viaja en ambos sentidos por el cable hasta que encuentra el destino deseado.

Anillo: es una topología en bus pero con los extremos conectados. Los datos atraviesan el anillo en una única dirección y van atravesando cada uno de los nodos, por lo que si uno de ellos no funciona, la red tampoco.

Malla: se pueden distinguir dos tipos: completamente conectados, en la que todos los nodos están conectados entre ellos, y parcialmente conectados, en la que algunos nodos pueden estar conectados punto a punto y otros pueden tener varias conexiones.

Híbrida: combinan dos o más topologías. La más famosa es la topología de **árbol**, en la que se conectan varias topologías de estrella, asemejando la forma de un árbol.

Cadena: se conecta cada ordenador en serie con el siguiente. Cada ordenador repite el mensaje al siguiente ordenador si éste no es su destino. Si se cierra el circuito se crea una topología en anillo, mientras que si se deja abierto se denomina topología lineal.

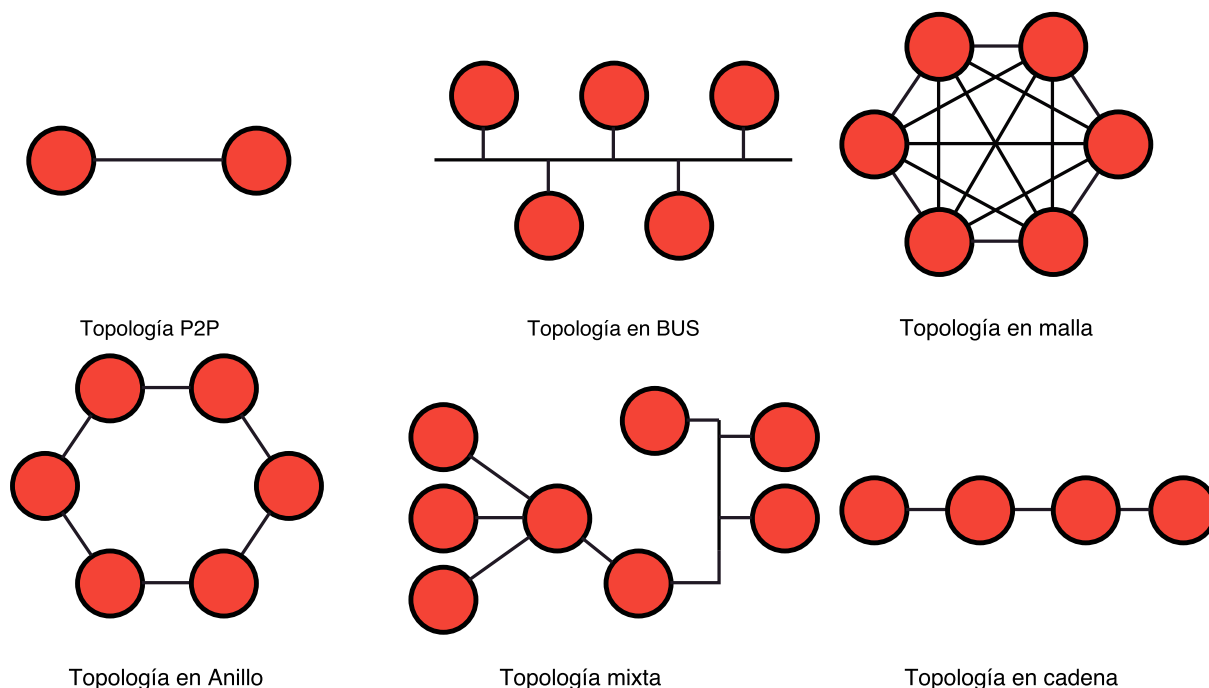


Figura 2.1: Representación esquemática de las diferentes topologías de red.

2.2.2 Nodos de una red

Router o enrutador: es un dispositivo de red que reenvía los paquetes analizando la capa 3 del modelo OSI (IP) y conecta dos redes.

Puente de red o bridge: Funciona en la capa 2 del modelo OSI. Es un dispositivo que conecta dos segmentos de red formando una única subred, por lo que las dos “redes” pueden conectarse e intercambiar datos sin necesidad de un *router*.

Conmutadores o switches: dispositivo de red que filtra los datagramas del nivel 2 OSI (*Data Link Layer*, ver 2.1, pág. 6), también conocidos como *frames*, y reenvía los paquetes recibidos entre los puertos, dependiendo de la dirección MAC de cada datagrama. La diferencia entre un conmutador y un concentrador es que el conmutador sólo reenvía los paquetes por el puerto necesario. También existen un tipo especial de conmutadores que pueden hacer uso de la capa 3 OSI.

Repetidores y concentradores: un repetidor es un dispositivo de red que, llegada una señal, limpia el ruido innecesario y la regenera. Un repetidor con múltiples puertos es un concentrador o *hub*, trabajan en la capa 1 del modelo OSI. Los repetidores requieren de un pequeño tiempo para regenerar la señal, lo que puede crear un retardo en el tiempo de envío de la señal.

Interfaces de Red: también conocido como tarjeta de red o *Network Interface Controller* (NIC), es un hardware, normalmente integrado en la placa base, que permite al ordenador conectarse a una red. Recibe el tráfico de una dirección de red. En las redes que hacen uso de Ethernet, tiene una dirección de Control de Acceso al Medio (MAC) única. Estas direcciones son administradas por el Instituto de Ingeniería Eléctrica y Electrónica (IEEE) evitando la duplicidad de estas. Cada dirección MAC ocupa 6 octetos, o 48 bits, a lo que suele ser representada como una cadena hexadecimal, por ejemplo: "4C:33:31:64:59".

Módem: Dispositivos que transforman señales analógicas a digitales y viceversa. Son usados mayoritariamente en el ADSL.

Cortafuegos o *firewalls*: dispositivo que controla la seguridad mediante reglas de acceso. Aceptan determinados paquetes mientras rechazan otros. En una red doméstica, se puede poner un firewall que sólo acepte tráfico de los puertos de uso común (Páginas Web, e-mail, etc.) y rechace otros más peligrosos (Acceso remoto, SSH, SMTP, SOCKS...).

2.3 Paquetes de red

Son cada serie de bits en los que se divide la información enviada por una red.

Según el modelo OSI, un paquete es estrictamente el PDU de la capa de red. El paquete de red se encuentra encapsulado en la capa anterior del modelo OSI. Por ejemplo, en estándares de comunicación TCP/IP, un segmento TCP puede ser llevado por varios paquetes IP transportados por varios frames de Ethernet. Es parecido a las unidades lingüísticas.

Está formado por varios protocolos y en él se distinguen tres partes:

Header o cabecera: Datos e información sobre el paquete. (Dirección IP, MAC, etc)

Payload o carga: Los datos que se quieren transferir.

Trailer o cola: En ocasiones es inexistente (como en UDP) pero suele ser un código de comprobación de errores.

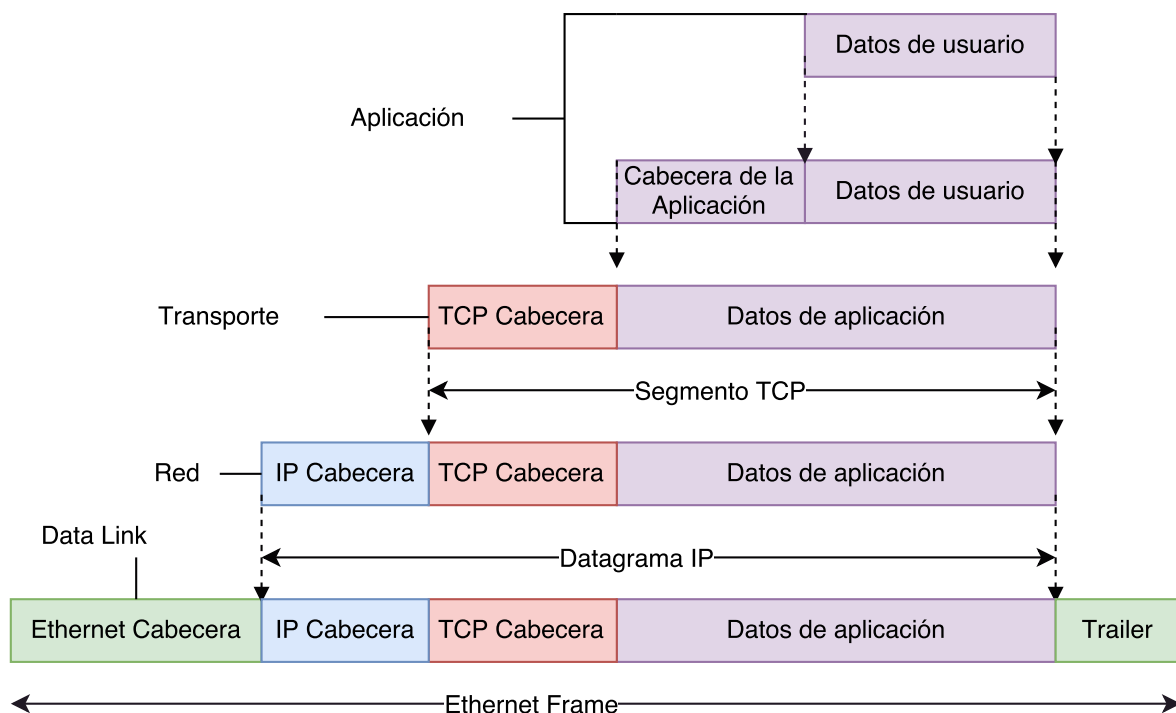


Figura 2.2: Encapsulación de red. El Datagrama IP es lo considerado 'Paquete de red'

2.4 Protocolos

Un protocolo de comunicación es un conjunto de reglas para intercambiar información entre enlaces de red. En una pila de protocolos, cada protocolo cubre los servicios del protocolo de la capa anterior. Por ejemplo, un e-mail se envía mediante el protocolo POP3 (*Post Office Protocol*, Protocolo de Oficina Postal) en la capa de Aplicación, sobre TCP en la capa de transporte, sobre IP en la capa de Red, sobre Ethernet para la capa *Data Link*, que está formado por bits. Para entenderlo mejor, es como la gramática de la lengua. Un sustantivo forma parte de un sintagma nominal, que forma parte de un sujeto, que a su vez forma parte de una oración. Siendo las ondas sonoras producidas la capa física y fonemas los bits.

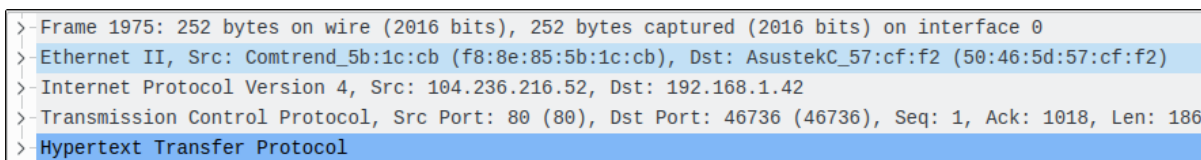


Figura 2.3: Captura de pantalla de Wireshark (Véase 3.1.6, pg. 21) en la que se muestran los protocolos que forman un paquete de red HTTP.

2.4.1 Familia de protocolos de internet

También conocido como *Internet Protocol Suite*, y más conocido como TCP/IP, es el fundamento de las redes informáticas. Se trata de un conjunto de más de 100 protocolos que permiten la conexión de ordenadores tanto en Internet como en LAN, incluyendo protocolos de las aplicaciones más usadas.

2.4.1.1 Aplicación

Es la capa en la que se envían los datos a otras aplicaciones en otro ordenador o en el mismo. Las aplicaciones hacen uso de las capas inferiores para asegurarse que los datos lleguen a su destino. Algunos de los protocolos más usados son:

- **HTTP** *Hypertext Transfer Protocol*: Protocolo de Transferencia de Hipertexto. Es el protocolo base de la World Wide Web. Se trata de texto estructurado que usa hiperenlaces entre nodos que también contienen texto. El cliente, al entrar en una URL (*Uniform Resource Identifier*, Identificador de Recursos Uniforme) a través de el agente de usuario (navegador) envía al servidor una petición de la página web, mediante HTTP. El servidor, envía como respuesta un documento HTML u otro recurso.

- **DNS** *Domain Name System*: Sistema de Nombres de Dominio. Un servidor DNS almacena una base de datos distribuida y jerárquica con información sobre el nombre del dominio y la dirección IP a la que está vinculada. Al intentar conectar a `ddavo.me`, el cliente pregunta al servidor cual es la dirección IP asociada a esa dirección, y se conecta a tal IP, en este caso 37.152.88.18. Para evitar tener que consultar continuamente con el servidor, se almacenan en una caché en el cliente.
- **TLS/SSL** *Transport Layer Security*, y su predecesor *Secure Sockets Layer* (Ver 2.5).
- **HTTPS** HTTP Seguro. Es HTTP con TLS aplicado.
- **DHCP** *Dynamic Host Configuration Protocol*: Protocolo de configuración dinámica del host. Este protocolo es controlado por un servidor DHCP que envía parámetros de configuración automática a los clientes. El ejemplo más común es el de cualquier Router doméstico, que asigna automáticamente a cada dispositivo una dirección IP diferente, pero dejando un rango en el que se pueden establecer IP's estáticas.
- **FTP** *File Transfer Protocol*: Protocolo de Transferencia de Archivos, te permite enviar archivos entre un cliente y un servidor. El protocolo TLS aplicado a FTP se denomina FTPS. Te permite acceder, mediante un usuario y contraseña, o de forma anónima, a un sistema de archivos jerárquico con nombres de archivo codificados. Utiliza el puerto 21 de forma predeterminada.
- **SSH** *Secure Shell*: Terminal seguro. Es un protocolo de red criptográfico que permite a un cliente conectarse a un servidor y ejecutar comandos de terminal como un usuario (conociendo el usuario y contraseña). Además, permite la creación de túneles, lo que permite asegurar cualquier aplicación a través de SSH, y el acceso a puertos bloqueados por el cortafuegos en el cliente. La mayoría de servidores de SSH incluyen un servidor de SFTP, el protocolo FTP con SSH aplicado.
- **IMAP** *Internet Message Access Protocol*: Protocolo de acceso a mensajes de Internet. Usa una conexión TCP/IP para conectarse a un servidor de e-mail y ver el contenido de los mensajes, sin necesidad de descargarlos. A diferencia de POP, te permite usar una bandeja de entrada desde varios clientes.

2.4.1.2 Transporte

Se encapsulan los datos de aplicación en un segmento o datagrama, dependiendo si el protocolo usado es TCP o UDP. Se encarga de transportar los datos por una red independientemente de la topología física.

- **TCP** *Transmission Control Protocol*: Protocolo de Control de Transmisión. Se aplica a los paquetes para administrarles un orden y un sistema de comprobación de errores. Con todas las funcionalidades, ocupa bastante espacio, lo que aumenta la latencia, aunque es más fiable para el envío de la mayoría de los datos.
- **UDP** *User Datagram Protocol*: Es un protocolo muy minimalista. A diferencia del TCP, no garantiza que los paquetes lleguen, o lleguen en orden, o protección ante duplicados. Reduce mucho la latencia ya que no usa *handshaking*. Por ello es usado por ejemplo para *streamings* de televisión o videollamadas.

2.4.1.3 Red

El objetivo es que los datos lleguen del origen al destino, aún cuando no están conectados directamente. Los enrutadores o *routers* son los dispositivos que cumplen esta función.

- **IP** *Internet Protocol*: Protocolo de Internet. Envía datagramas o paquetes de red a través de redes. Tiene una función de enrutamiento que es la que permite la interconexión de redes, y la existencia de Internet. Es un protocolo que encapsula el paquete definiendo en el *header* (cabecera) las direcciones IP del servidor y el cliente, o remitente y destinatario. La versión usada actualmente es IPv4 desarrollado en 1981, pero poco a poco se va abriendo paso la versión IPv6. La mayor diferencia es que la versión cuatro cuenta con direcciones de 32 bits lo que permite tan sólo unos 4.3 millardos (2^{32}) de direcciones, mientras que la versión 6 tiene direcciones de 128 bits, lo que permite más de 340 sextillones (2^{128}) de direcciones
- **ICMP** *Internect Control Message Protocol*: Es un protocolo que no es usado por aplicaciones de usuario (a excepción de herramientas de diagnóstico como ping o traceroute). Lo usan los dispositivos de red, como los routers, para enviar notificaciones o mensajes de error indicando que un servicio no está disponible.

2.4.1.4 Link

Capa encargada del acceso al medio físico de la red. También cumple otras funciones como incluir una comprobación de errores e identificar cada dispositivo de forma única.

- **ARP** *Address Resolution Protocol*: Protocolo de resolución de direcciones. Es un protocolo que convierte direcciones de la capa de Red a la capa de Enlace (dir. IP a dir. MAC). El dispositivo, al conectarse a una red, envía un *frame* ARP con su dirección MAC y su IP, para que los demás dispositivos de la red lo almacenen en su memoria y poder usar ambas direcciones al enviar un paquete.
- **MAC** *Media access control*: Control de acceso al medio. Es un conjunto de protocolos (Como Ethernet o IEEE 802.11 [WiFi]) encargados de asignar el medio físico de la red. Evita colisiones entre paquetes asegurándose de que el medio está libre y evitando así la transmisión simultánea.

2.5 Seguridad de redes

Consiste en el conjunto de acciones que toma el administrador de redes para prevenir y evitar acceso no autorizado, mal uso, o caída del servicio de red.

2.5.1 Tipos de ataques

Hay dos tipos de ataques de red, los activos y los pasivos. Son ataques pasivos cuando el intruso intercepta los datos que viajan por la red, y se considera activo cuando el atacante modifica el funcionamiento normal de la red.

Algunos de los ejemplos de los ataques más comunes son:

- **Ataques pasivos**
 - **Sniffing o analizador de paquetes**: Mediante un software se muestran los datos de los paquetes de red enviados y recibidos por la red.
 - **Escáner de puertos**: Se envían numerosas peticiones al servidor por los puertos más comunes, así se comprueba que puertos están abiertos. Por ello es recomendable cambiar los puertos por defecto de los servidores importantes.

–**Escáner IDLE:** Se realiza un escáner de puertos para saber que servicios están disponibles, pero a través de otro ordenador “zombie”, y observando el comportamiento de éste.

- **Ataques activos**

–**Ataque de Denegación de Servicio:** Se “desborda” el ancho de banda mediante el envío de muchas peticiones a un servidor, además de ser de un tamaño excesivo.

–**Ataque DDoS:** *Distributed Denial of Service*, o un ataque de Denegación de Servicio distribuido. Varios ordenadores hacen un ataque DoS a un mismo servidor, algunas veces los ordenadores forman parte de una botnet, y en ocasiones ocurre sin querer (al haber demasiado tráfico de red).

–**Phishing:** Con el objetivo de obtener información como nombres de usuario y contraseña o tarjetas de crédito, se crea una página de apariencia parecida a la página que trata de simular. Los usuarios más incautos no notarán el cambio e introducirán sus datos en esta página.

–**SQL Injection:** Es una técnica de inserción de código. Al pedir un servidor SQL datos como “Nombre” o “Apellido”, se introduce junto a estos código malicioso que el servidor puede ejecutar. Por ejemplo, `SELECT * FROM alumnos WHERE nombre = '<nombreintroducido>'` ;. <nombreintroducido> puede ser Pablo o Juan, pero si se introduce `x'` ; `DROP TABLE alumnos;` `SELECT * FROM asignaturas WHERE 't' = 't'`, el código que interpreta el servidor eliminaría la tabla alumnos por completo.

–**Ataque Smurf:** Es una especie de ataque DDoS. Se envían paquetes ICMP (probablemente pings) a distintas máquinas, pero estos paquetes que se envían, el valor de la dirección IP del remitente es la dirección IP del objetivo al que se quiere atacar. Por lo que, las máquinas a las que se las ha enviado el mensaje ICMP responderán todas al objetivo, haciendo así un DDoS.

–**DNS poisoning:** Se modifica la caché de DNS de un ordenador, redireccionando a una IP incorrecta, de esta manera se puede realizar un ataque de phishing sin que lo sepa el usuario del ordenador. En el caso de hacerlo con las tablas de ARP, se denomina *ARP Poisoning*.

2.5.2 Contramedidas

Acciones que se pueden tomar para evitar algunos de los ataques de red más comunes.

2.5.2.1 Encriptación

Se suele denominar también E2EE o *End-to-end encryption*, es decir, encriptación de punto a punto.

Se suelen usar claves PGP (*Pretty Good Privacy*, Privacidad bastante buena) para cifrar correos electrónicos y otros archivos. Para HTTP lo más común es la encriptación TLS, aunque también se está utilizando actualmente para email. El servidor genera o contiene una clave o certificado, luego el cliente, debe recibir o tener esa clave para poder desencriptar el mensaje.

2.5.2.2 Cortafuegos

Primero necesitamos definir lo que es un **puerto**. Un puerto es un punto final de comunicación en un Sistema Operativo. El puerto siempre está asociado a una dirección IP y a un tipo de protocolo. Así completa el origen o destino de un paquete de red. Se aplica en la capa de transporte del modelo OSI. El puerto es un número de 16 bits, por lo que será un número comprendido entre 0 y 65536. Multitud de puertos están ya reservados por diversos protocolos y programas, como el 80 para HTTP, 22 para SSH o 25 para SMTP.

Un cortafuegos es un software que supervisa el tráfico de entrada y salida de datos, basado en unas reglas. Si un paquete de red cumple esas reglas, es rechazado. Pueden bloquear un paquete destinado a un puerto, de un protocolo (Bloquear SSH de Internet, pero no local), de una IP específica, entre otros atributos. También pueden configurarse en modo negativo o whitelist, aceptando tan sólo los paquetes que cumplan las reglas. Por ejemplo, puedes especificar que no acepte tráfico en el puerto 23. Pero igualmente puedes especificar que sólo acepte tráfico en el puerto 23.

Capítulo 3

El simulador de redes

La parte práctica del Proyecto de Investigación, el simulador de redes de nombre *InvProy*, ha sido la parte más extensa del proyecto, que más tiempo, esfuerzo y recursos ha ocupado. Se podría decir que el proyecto entero es la parte práctica. Como curiosidad, el nombre de *InvProy* viene del acrónimo formado mediante la permutación de las palabras “Proyecto de Investigación”, quedando “*Investigación de Proyecto*” y de ahí *InvProy*.

3.1 Herramientas usadas en la creación del software

Todo el software que se ha usado para la creación de este programa, es software libre, debido a las ventajas citadas anteriormente. A continuación, se listan las herramientas que se han usado para la creación tanto del programa como de este documento.

3.1.1 GNU/Linux

También llamado incorrectamente sólo Linux, es una manera de llamar al Sistema Operativo (OS) combinación del kernel Linux (Basado en Unix) y el OS GNU (Acrónimo recursivo *GNU's Not Unix*, o GNU no es Unix). Es el gran ejemplo por excelencia del Software Libre. Es el sistema operativo más utilizado, pues es usado en la mayoría de los servidores, y además, otros sistemas operativos como Android están basados en éste. Puedes instalar Linux desde el código fuente o instalar distribuciones o *distros*.

3.1.1.1 Distros

Son las distintas distribuciones de software de GNU/Linux. Es decir, un conjunto de software preconfigurado y compilado formado por el Sistema Operativo GNU, el kernel de Linux y otros tantos paquetes, dependiendo de los usuarios a los que esté dirigida la distribución. Pueden crearse con el soporte de una empresa; como Ubuntu (Canonical Ltd.), openSUSE (Novell) o Fedora (Red Hat); y otras mantenidas por comunidades como Debian, Gentoo o Arch Linux.

Para el desarrollo de este proyecto he usado dos distros diferentes. Una llamada **Arch Linux**, que es *rolling release* (No tiene "versiones", sino que siempre se va actualizando con los últimos paquetes disponibles, por lo que siempre está actualizado) en la que se ha ido haciendo la programación, y **Ubuntu 16**, basado en Debian, por lo que está bastante menos actualizado y se han tenido que hacer correcciones en el programa para que pueda funcionar con versiones más antiguas de las dependencias. Se ha usado Ubuntu para comprobar el funcionamiento del software, ya que es la distribución más usada en los hogares y en educación.

3.1.2 Git y Github

Git es un software diseñado por Linus Torvalds con el que se puede crear un Sistema de Control de Versiones (VCS [Version Control System]). Este programa te permite de forma sencilla volver a una versión o *commit* anterior del programa, así como enviarlas a un repositorio remoto e incluso publicarlas en línea. Su punto fuerte son las *branches* o "ramificaciones" del código, haciendo que la rama *master* (principal) siempre sea funcional. Para ello creamos una nueva rama para cada nueva funcionalidad del programa. La implementación del nuevo código a otra rama se denomina *merge*. Otra de las funcionalidades que implementa es *clone*, que te permite descargar un proyecto si tienes la URL del repositorio git.

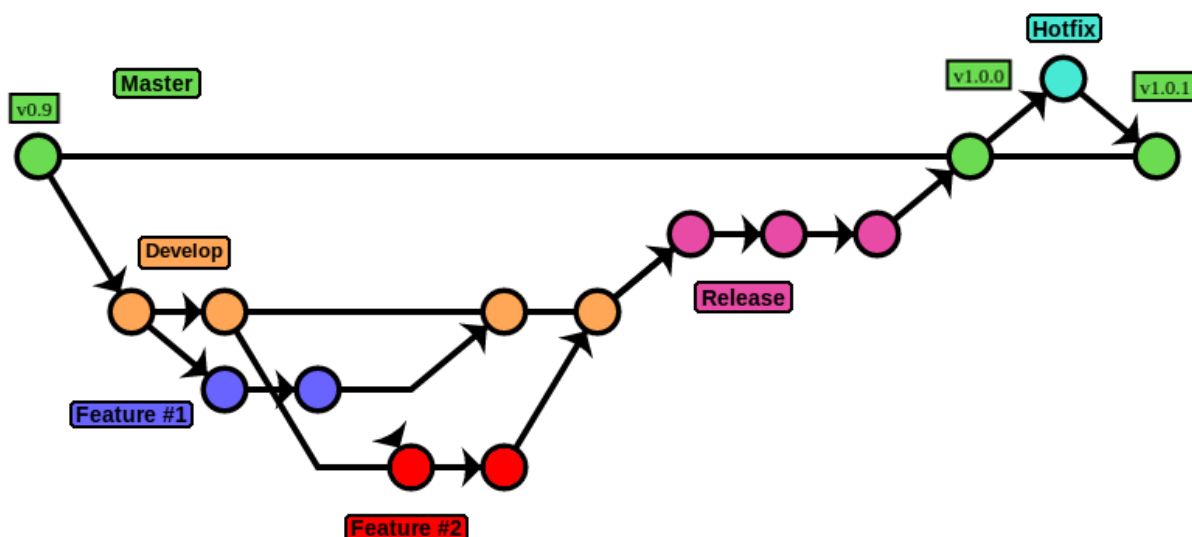


Figura 3.1: *Gitflow* o flujo de trabajo de Git

Para usar Git, se suele recomendar seguir un *Git workflow* o flujo de trabajo de Git, en ocasiones denominado *gitflow*. El más común es el basado en 4 nuevas ramas, aparte de master.

Develop: es la rama de desarrollo. Se van aplicando las nuevas funcionalidades a esta rama, para luego convergerlas en la rama Release que se va a publicar.

Release: una vez hayamos terminado en la rama de desarrollo, se converge Develop con Release y se procede a solucionar los bugs que se vayan descubriendo. Cuando se hayan solucionado todos los bugs y la siguiente versión del programa esté disponible para el público, se hace merge en Develop y en Master, además de aplicarle al commit una etiqueta con el nombre de la versión. (2.2.1, por ejemplo).

Hotfix: Es una rama dedicada a solventar los bugs que un usuario descubra en una versión ya lanzada de la aplicación. Cuando un usuario descubre un bug, se crea una nueva rama a partir de la última versión de master, se soluciona el bug en esa rama y luego se vuelve a hacer merge en master y develop.

Feature <x>: Donde <x>el nombre de la funcionalidad. Es una rama dedicada a una nueva funcionalidad, se crea a partir de Develop, y una vez terminada, se hace merge en Develop de nuevo.

GitHub es una plataforma de desarrollo colaborativo que te permite alojar tus repositorios Git. Su uso es gratuito si el código almacenado es público. Además, te permite tener una wiki y una página web para tu proyecto, junto a otras funciones. Una de sus funciones estrella es la visualización online del repositorio, con la que cualquier persona tiene acceso al código y los archivos antes de descargarlos. Otra función útil es el apartado de *Issues*, en el que los usuarios de tu código pueden reportar los bugs del programa o aportar nuevas ideas en forma de "foro". Tanto el programa como este documento están disponibles en GitHub en los siguientes enlaces. <https://github.com/daviddavo/InvProy> y <https://github.com/daviddavo/InvProy-text>

3.1.3 LaTeX

L^AT_EX o, en texto plano, LaTeX, pronunciado con la letra griega Ji (X), es un software libre orientado a la creación de textos escritos comparable a la calidad tipográfica de las editoriales. Mediante la importación de paquetes y comandos o macros se puede dar formato al texto al igual que con cualquier otro editor, exportándolo posteriormente a PostScript o PDF. Está orientado a documentos técnicos y científicos por su facilidad a la hora de incluir fórmulas o código e importar paquetes que cumplan las necesidades de los usuarios. No es un procesador de textos, pues está más enfocado en el contenido del documento que en la apariencia de éste. El código del documento puede ser editado con cualquier editor de texto plano como *nano* o *emacs*, aunque he usado una IDE llamada **texmaker**.

3.1.4 Python

Es un lenguaje de programación interpretado (sólo se traduce el programa a código máquina cuando se debe ejecutar esa parte del código, por lo que no hace falta compilarlo) que destaca porque sus programas poseen una sintaxis más legible que la de el resto de lenguajes. Soporta tanto programación imperativa como programación orientada a objetos. Usa variables dinámicas, es multiplataforma, y, además, es de código abierto, lo que permite distribuir el programa en Windows al distribuir los binarios de Python junto a él. En este proyecto, la versión de Python utilizada es la 3.4 en adelante.

3.1.5 Gtk+

Es un conjunto de bibliotecas o librerías (conjunto de funciones y clases ya definidas preparadas para el uso de los programadores) desarrollado por la GNOME foundation destinado a la creación de Interfaces Gráficas de Usuario (GUI), también, al igual que Linux forma parte del proyecto GNU.

Contiene las bibliotecas de GTK, GDK, ATK, Glib, Pango y Cairo; de las que he usado fundamentalmente GTK para crear la interfaz principal del programa; **GDK** al utilizarlo como intermediario entre los gráficos de bajo nivel y alto nivel y **Cairo** para la creación de algunos de los elementos gráficos del programa.

Al utilizar este conjunto de librerías, se ha conseguido que sólo sea necesario descargar una dependencia del programa, que además suele venir instalada en la mayoría de distros de Linux. Por ejemplo en una instalación limpia de Ubuntu 16 (sin descargar paquetes adicionales)

el programa funciona perfectamente. Para usarlo en Python se ha tenido que importar la librería de PyGtk, que también suele venir incluida en la distribución.

3.1.6 Wireshark

Wireshark es un *packet sniffer* o analizador de paquetes; te muestra los paquetes de red reales enviados y recibidos por una tarjeta de red, lo que facilita la creación del simulador de redes. También te separa las distintas partes de la encapsulación del paquete y además te permite buscar entre los paquetes de red añadidos y recibidos, pudiendo añadir filtros de búsqueda para los distintos campos del paquete y para las distintas capas.

3.2 Instalación

3.2.1 Ubuntu / Debian

Tan sólo se debe descargar el paquete del programa. Para ello debemos usar apt-get:

```
~ $ sudo apt-get install invproy
```

En caso de no estar en los repositorios, hay que hacerlo manualmente. Descarga el paquete de <https://github.com/daviddavo/InvProy/releases/latest>. Una vez descargado, abre una terminal donde se haya descargado el paquete e instálelo.

```
Descargas $ sudo dpkg -i invproy_x.y.z_all.deb
```

Donde 'x', 'y', y 'z' son la versión del paquete descargado. Para iniciar el programa debes usar la lista de programas de tu escritorio.

3.2.2 Arch Linux

Descarga la versión más reciente para Arch Linux de <https://github.com/daviddavo/InvProy/releases/latest>. Una vez descargado, abre una terminal donde se haya descargado el paquete e instálalo.

```
~ $ sudo pacman -S base-devel #Lo necesitas para compilar el paquete
#Ahora elige el sitio donde descargaras el paquete. Aqui no se va a instalar.
~ $ cd Builds
Builds $ curl -O <url> #Lo descargamos
Builds $ tar -xvzf invproy.tar.gz
Builds $ cd invproy
Invproy $ makepkg -sri
```

Y ya lo tendrías instalado en tu ordenador.

3.2.3 Ejecución manual / instalación portable

Lo primero que necesitará es descargar las dependencias. Esto depende del Sistema Operativo. En el caso de GNU/Linux, sólo es necesario descargar python3-gobject. Después, clonamos el repositorio de git. Ejemplo en Ubuntu:

```
~ $ sudo apt-get update && sudo apt-get upgrade
~ $ sudo apt-get install git python3-gobject
~ $ cd Descargas
Descargas $ git clone https://github.com/daviddavo/InvProy.git
```

Una vez ya tenemos el repositorio de git clonado:

```
Descargas $ cd InvProy
Descargas $ python3 Main.py
```

En el caso de querer usar el programa desde una interfaz gráfica, vamos con nuestro explorador de archivos a la carpeta donde queramos descargarlo. Abrimos una terminal y descargamos el programa con `git clone https://github.com/daviddavo/InvProy.git`. Luego entramos en la carpeta y ejecutamos el archivo `Main.py`

3.3 Uso del programa

Esta guía ha sido creada usando la versión v0.2.3-alpha, por lo que en algunos apartados pueden haberse realizado cambios en versiones posteriores.

El programa está siendo diseñado para tener una mayor facilidad de uso, pensando en una interfaz intuitiva y simple que pueda ser utilizada sin la necesidad de ningún apoyo externo al programa (instrucciones, documentación, tutor). Para ello, en la versión v0.4 será añadido un asistente o 'tutorial' que guíe a los alumnos la primera vez que se acceda al programa; además de añadir más información, dentro de la interfaz, sobre redes.

A continuación, se incluye una captura de pantalla de la interfaz de InvProy Alpha, explicando el funcionamiento de los distintos botones de la interfaz.

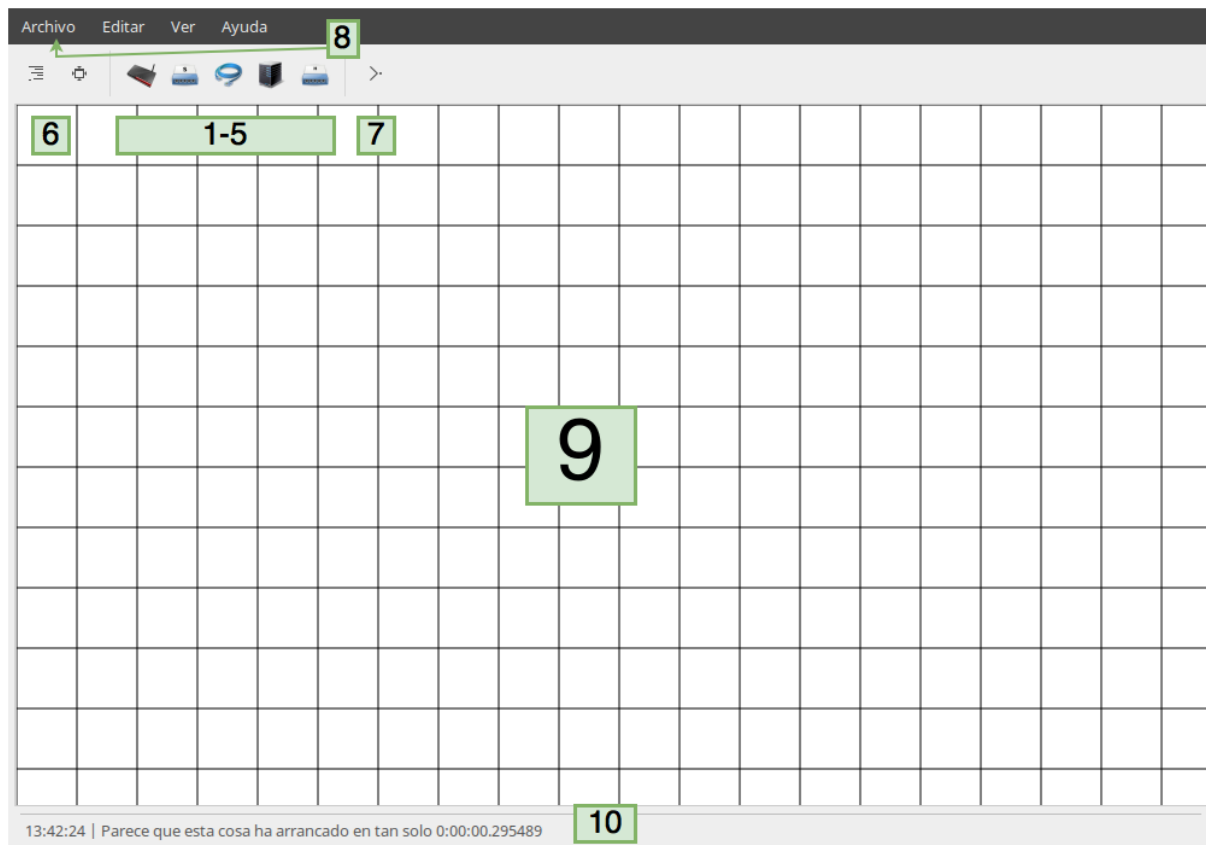


Figura 3.2: Interfaz de InvProy Alpha. Al usar Gtk+, los temas se pueden cambiar, así que la apariencia del programa puede ser distinta dependiendo del tema de escritorio que estés usando.

- 1-5. También se puede activar con las letras Q, W, E, R, T; respectivamente. Los botones, te permiten (de izquierda a derecha): colocar un router, colocar un switch, conectar dos objetos, colocar un ordenador y colocar un hub. Para ello primero haces click en el botón y luego haces click en el lugar donde quieras colocar el objeto. En el caso de los cables debes hacer dos clicks, uno en cada objeto a conectar.
6. Abre el menú de “Información de dispositivos”, que proporciona información como la dirección IP y MAC, el nombre, o los dispositivos a los que está conectado. (Ver figura 3.3)
7. Te permite enviar un ping de un ordenador a otro (El botón funciona a partir de v0.3).
8. Abre el menú de archivo, en el que puedes cargar un archivo, crear uno nuevo, guardarlo, y cerrar el programa.
9. Es la ventana donde puedes colocar los objetos. Puedes moverte a través de ella y en el menú de ‘Ver’ puedes cambiar el que se vea la rejilla de fondo.
10. Aquí se encuentra una barra con información sobre el funcionamiento actual del programa.

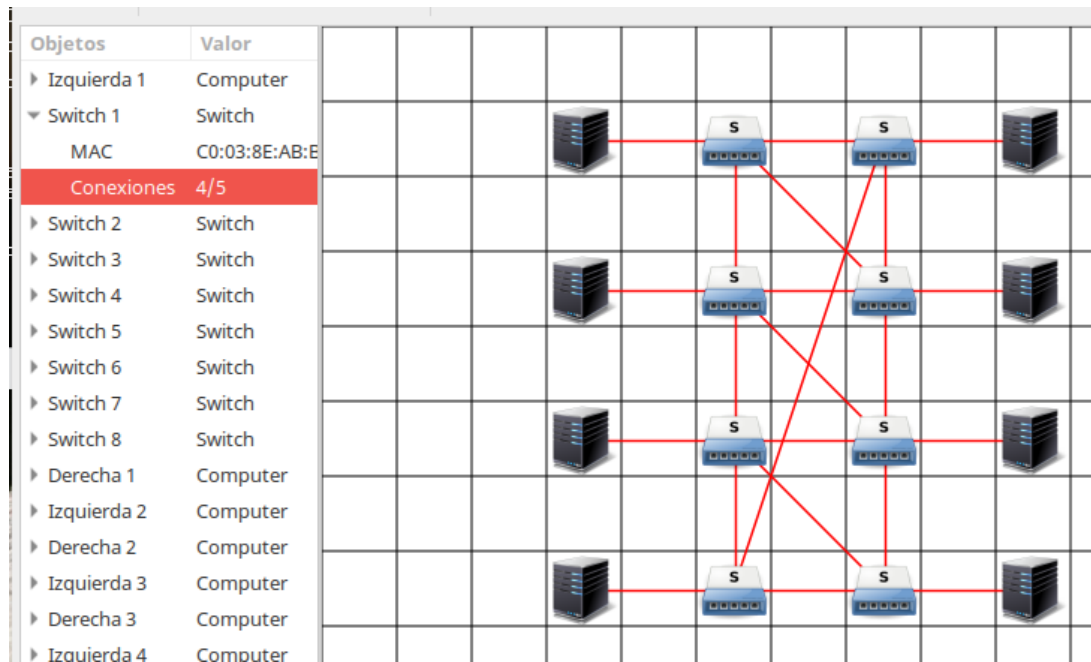


Figura 3.3: Menú de **Información de Dispositivos** junto a una red de topología de malla

Para incluir un objeto en la rejilla, se hace click en el icono del objeto y luego en el lugar donde se quiera poner. Cuando tenemos dos objetos podemos conectarlos si hacemos click primero en el icono del cable, luego en un objeto y después en otro.

3.3.1 Configuración

Al no haber una ventana de configuración del programa, la configuración debe hacerse de forma manual editando el archivo `Config.ini` (Ver D.4). Este es un archivo de texto sin formato en el que se le asigna un valor a cada variable.

`wres` y `hres`: El tamaño (en píxeles) del ancho y el alto de la ventana principal.

`viewport-sqres`: El tamaño en píxeles del lado de los cuadrados de la rejilla.

`viewport-wres` y `viewport-hres`: El número de cuadrados que tendrá de alto y de ancho la rejilla.

`cable-color`: Color por defecto de los cables en HTML.

`start-centered`: Al iniciar el programa, iniciar en el centro de la rejilla en lugar de arriba a la izquierda.

`revealer-show-default`: (True o False). Mostrar por defecto la ventana con la información sobre los dispositivos.

`respack`: Directorio del "Pack de recursos"

routing-ttl: Tiempo de vida en segundos de las entradas en la tabla de redireccionamiento de los switches.

def-max-connections: Conexiones máximas por defecto de un conmutador/concentrador.

3.3.2 Ejemplo: Envío de Ping entre dos dispositivos

Lo primero que vamos a hacer, es colocar un Switch al que poder conectar los ordenadores. Después colocamos y conectamos hasta 5 ordenadores (es el máximo de conexiones por defecto) al Switch. Después, para cada ordenador, hacemos click derecho y en el menú emergente pulsamos “Editar Objeto”, con lo que se abrirá una ventana como la de la Figura B.5. Aquí podemos asignar una dirección IP al ordenador. Tras asignar 5 direcciones IP diferentes a los ordenadores, en cualquiera de ellos hacemos click derecho y clickamos en “Ping”, haciendo que aparezca una ventana con un cuadro de texto como en Fig. B.3. En este cuadro introducimos la dirección IP del dispositivo al que queremos enviar el Ping y pulsamos la tecla Intro, así el ordenador enviará un paquete Ping Request a la IP especificada. Cuando el paquete llega al equipo con esa dirección IP, este responderá con un paquete parecido, pero en este caso será un ping de respuesta (simbolizado en rojo), por lo que el destino será el primer ordenador. En el caso de que se produzcan cambios en la red mientras el paquete viaja por esta, el paquete dispone de un tiempo de vida, por lo que cuando llega a 0 se destruye.

3.4 Funcionamiento del programa

Se ha creado haciendo uso de todas las herramientas anteriormente mencionadas.

El programa posee distintas clases. Se pueden diferenciar en cuatro tipos: Clases de Interfaz (`MainClase`, `w_changethings...`), Clases de Dispositivos (`ObjetoBase`, `Switch`, `Computador`), Clases de Red (`packet`, `frame`) y clases de apoyo (`MAC`, `IP`, `Port`, `Cable`).

Todas las clases poseen, como mínimo, una función llamada `__init__`, que es la encargada de crear el objeto y establecer las variables más importantes (Coordenadas, variables vacías, dirección MAC...).

3.4.1 Main.py

Es el archivo principal del programa. Contiene las funciones más importantes, además de las clases para crear los objetos. Primero trata de importar los módulos necesarios, comprobando uno a uno si están instalados, y advirtiéndolo al usuario en el caso de que no estén instalados.

3.4.1.1 MainClase

Es la clase principal de la interfaz del programa. Se encarga de administrar la ventana principal de la interfaz.

Posee varias funciones como `on_key_press_event` y `on_key_release_event` (370:416), que actúan cada vez que se pulsa una tecla y se encargan de hacer las acciones necesarias para esa tecla (o combinación de teclas). Otra función importante es `toolbutton_clicked`, que se acciona cada vez que se pulsa un botón (de arriba) y se encarga de comunicarlo a la rejilla.

También contiene una subclase, llamada `ObjLst` (263:332¹), encargada de la lista de objetos de la parte izquierda de la interfaz.

¹Notación para escribir la ubicación del código. Línea inicial:Línea final@Archivo. Si se omite el nombre de archivo, es porque ha sido anteriormente mencionado.

3.4.1.2 Grid

Es la clase de la rejilla, se desarrolla de la línea 498 hasta la 677. Tiene varias 'capas', una para los cables, otra para el fondo, otra para los dispositivos... Así, en el caso de que dos elementos se solapen, los dispositivos siempre permanecerán al frente del fondo y los cables. El fondo se hace creando una línea horizontal cada `sq` píxeles, y otras tantas verticales del mismo modo, siendo `sq` el parámetro `viewport-sqres` del archivo `Config.ini`.

clicked_on_grid 578:639 Función que se encarga de realizar distintas acciones dependiendo de dónde se haya hecho click dentro de la rejilla. Para ello, primero debe comprobar si ahí hay o no un objeto, y una vez lo ha comprobado, comprobar si uno de los botones para colocar un objeto ha sido pulsado.

gridparser 641:651: Es una función muy sencilla. Te convierte coordenadas de la rejilla, a coordenadas en píxeles (usadas por Gtk).

resizetogrid 653:657: Otra función sencilla. Dada una imagen, la convierte al tamaño de un cuadrado de la rejilla.

searchforobject 659:672: Encargada de comprobar si, dadas unas coordenadas, hay un objeto en estas.

moveto 563:576: Te mueve una imagen dada a unas coordenadas dadas. En el caso de que de que no esté en la rejilla, crea la imagen, y en el caso de que ya esté en ella, la mueve al lugar designado.

3.4.1.3 ObjetoBase

En Python, existe la herencia de clases. Esto quiere decir que una clase puede heredar las funciones y los atributos de otra, en forma de cascada. La clase principal de la que heredan el resto de dispositivos de red es **ObjetoBase**. Algunas de sus funciones son estas:

- **compcon**: Es una función que poseen todos los dispositivos de red, que dado un objeto **Computador**, retorna todos los ordenadores que están conectados a la misma red. Se encuentra en 822:864. Está formada por una lista que contendrá los ordenadores conectados y una función llamada **subcompcon** (827:847), que comprueba las conexiones del objeto, añadiendo la conexión a la lista si es un ordenador. En el caso de que sea un conmutador o un concentrador, llama a la función **subcompcon** con ese objeto como argumento, por lo que comprueba las conexiones de ese objeto y las añade a la lista del primero, entrando en un bucle hasta que ha comprobado toda la red. La función es usada por el programa cuando es necesario comprobar si dos ordenadores están conectados, por ejemplo. (Ver figura 3.4)
- **load**: 877:896 Al cargar un objeto de un archivo, hay determinadas propiedades del objeto que deben ser establecidas de cero, y determinadas funciones que deben ser llamadas, para ello existe esta función.
- **update**: Esta función, bastante importante se encarga de actualizar la información del objeto en la interfaz de usuario. Es llamada cada vez que se produce un cambio en el objeto; como al conectarlo, editar el nombre, o desconectarlo de otros objetos.
- **connect**: 898:941 Esta función se encarga de establecer las conexiones entre dos objetos.
- **disconnect**: 942:989. Realiza lo contrario de **connect**, desconecta un objeto de otro. O un objeto de todos a los que está conectado.
- **packet_received**: Esta es la función por defecto que se ejecuta cuando un dispositivo ha recibido un paquete. Todos los dispositivos de red tienen una función diferente que sobrescribe a esta (pues no es el mismo comportamiento el de un conmutador que el de un ordenador).

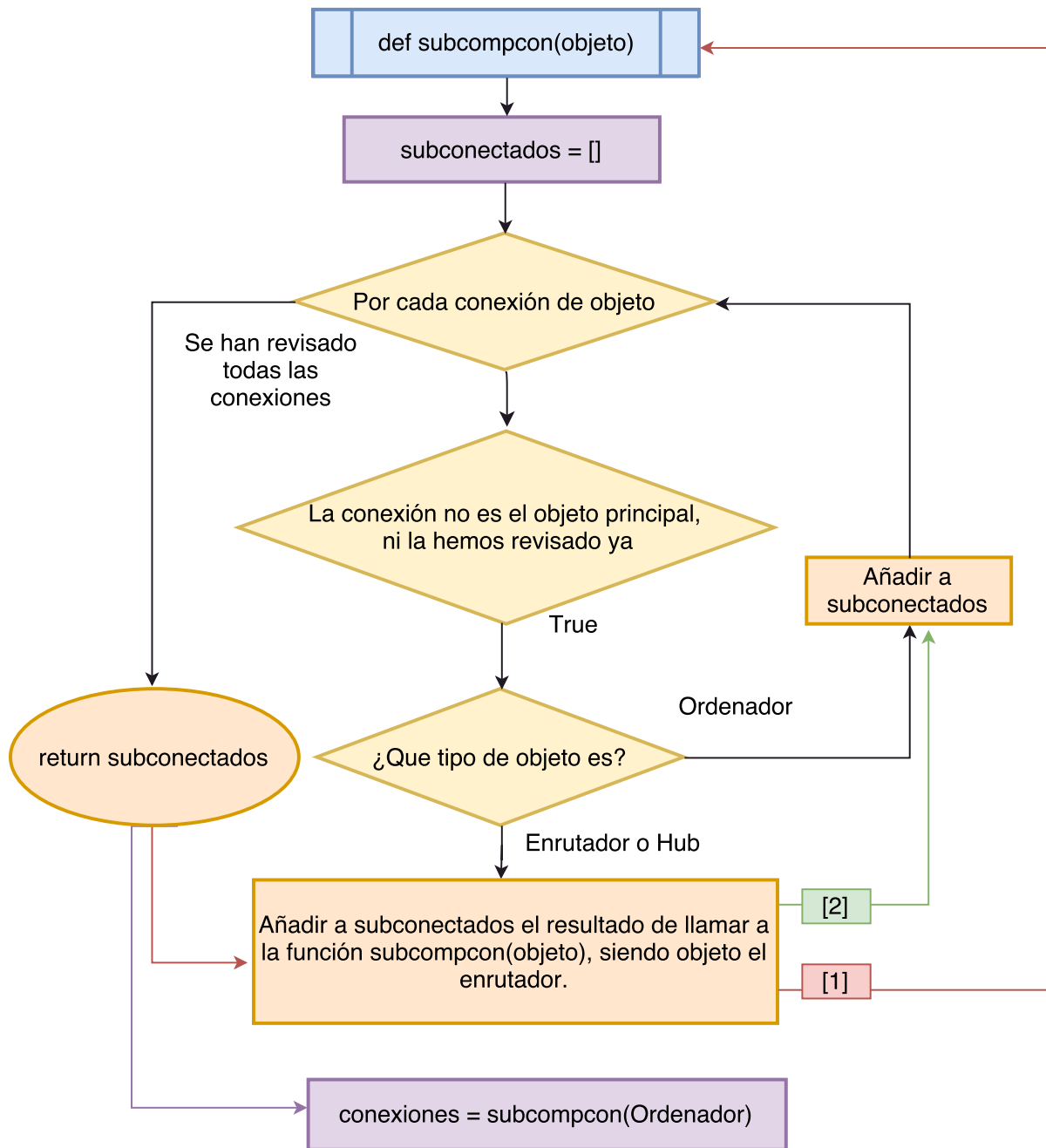


Figura 3.4: Diagrama de flujo del funcionamiento de la función `compcon`.

3.4.1.4 mac

Esta clase es la que crea los objetos que serán una dirección MAC. Transcurre de la línea 1024 a la 1059 y contiene varias funciones técnicas, pero la única importante es `genmac`, encargada de generar una dirección MAC aleatoria de 48 bits de longitud.

3.4.1.5 Port y w_switch_table

`Port` es una clase que usan los conmutadores y concentradores. Simula un puerto de red. Tan sólo posee cuatro funciones: `__init__`, que es la que se usa al crear el objeto; `connect`, para conectar un objeto al puerto; `disconnect`, para desconectarlo y `is_available`, para saber si el puerto está disponible u ocupado. 1080:1099

La clase `w_switch_table` es la encargada de la ventana de visualización de la tabla de enrutamiento del Switch. 1101:1160

3.4.1.6 Clases de paquetes de red

Ocupan entre el 25 % y el 30 % del código. Son clases como `packet` (la clase base), `eth` (paquete con *frame* aplicado), `icmp` (paquete con ICMP) y la última clase `Ping`, que hereda de `icmp` y se encarga de crear un paquete de red, bit a bit, dados una dirección IP de destino y de origen. Entre todas estas clases debemos destacar dos funciones:

- `animate` es una función que poseen todos los tipos de paquetes de red y se encarga de poner un paquete de red en la interfaz, y de que este se mueva. Para ello, hace una combinación de dos movimientos, uno en el eje x y otro en el eje y, la longitud que debe moverse en total la divide entre el número total de fotogramas y así consigue la distancia que debe moverse cada fotograma. Cuenta dentro con una subfunción, `iteration`, que se encarga de poner la imagen cada fotograma en su sitio y eliminar la imagen del fotograma anterior. Esta función ha sido posible gracias a los conocimientos sobre vectores adquiridos durante primero de Bachillerato. 1627:1717
- `create` es una función propiedad de `Ping`, que dadas una dirección IP de destino y origen, crea un paquete de red, bit a bit, basado en el modelo real de paquetes de Ping inspeccionado por Wireshark, y confirmado en libros de teoría. Es una función que, aunque parezca sencilla, fueron bastantes horas de trabajo, pues es bastante complejo tratar con bits.

```

1782 def create(r, sourceip, desti_ip, *n, payload=int( 4.3*10**19 ) << 6 | 42, \
1783         flags=0b010, ttl=32):
1784     self = Ping()
1785     if r == 0:
1786         Type = 8
1787         self.color = "#4CAF50"
1788     if r == 1:
1789         Type = 0
1790         self.color = "#F44336"
1791
1792     self.payload = payload
1793
1794     vihlto = 0b0100010100000000
1795     #20 Ipheader + 8 ICMPHeader + Payload
1796     lenght = int( 20 + 8 + ( int(math.log(payload, 2))+1)/8 ) #In Bytes
1797     frag_off = 0b00000000000000
1798     protocol = 1
1799     checksum = 0 #No es necesario porque no hay cables
1800     sourceip = int(sourceip)
1801     desti_ip = int(desti_ip)
1802     identific = Ping.identifi
1803     Ping.identifi += 1
1804
1805     self.ip_header = (((((((((vihlto << 16 | lenght)<<16 | identific) << 3 | flags) << 13 | frag_off) \
1806     << 8 | ttl) << 8 | protocol) << 16 | checksum) << 32 | sourceip) << 32 | desti_ip)
1807
1808     identifier = 1*2**15 + 42 * 2**8 + 42
1809     Code = 0
1810     icmp_header_checksum = random.getrandbits(16)
1811     self.icmp_header = (((((((Type << 8 | Code)<< 16 | checksum) << 16 | identifier) << 16 | identific)
1812     self.pck = icmp(self.ip_header, self.icmp_header, self.payload)
1813
1814     self.str = self.pck.str
1815     self.lenght = self.pck.lenght
1816     self.bits = self.pck.bits
1817
1818     return self

```

3.4.2 save.py

Es un archivo que se encarga de guardar y cargar archivos. Está compuesto por dos funciones: save y load, encargadas de guardar a un archivo y cargar a un archivo, respectivamente. Para ello, usan una librería nativa de Python llamada pickle, que se encarga de la serialización de los objetos, y la posterior deserialización de estos. Este método de serialización debería ser cambiado en una versión posterior, ya que no es retrocompatible, es decir, no te permite cargar archivos creados con una versión anterior del programa, además de que hace que los archivos de guardado sean demasiado pesados.

3.4.3 Interface.glade

Este archivo, de mil trescientas veinticinco líneas, es el encargado de establecer las propiedades de la interfaz. No ha podido ser incluido en el anexo debido a su larga extensión, ya que usa XML, un lenguaje que es bastante redundante, aunque sencillo de usar.

3.4.4 Dispositivos

Existen cuatro tipos dispositivos: los Computadores, que tienen la mayor programación; los Switches, que se encargan de manejar los paquetes de red; los Hubs, que son como los Switches, pero reenvían los paquetes por todos sus puertos y los Routers, que tan sólo existen de forma visual, pero no tienen ninguna función de momento. Por lo que sólo vamos a hablar de los Switches y los Ordenadores. Para cambiar los parámetros hay que hacer click derecho en el dispositivo al que se le deseen cambiar los parámetros y luego en la entrada de 'Editar objeto'. A lo que aparecerá una ventana como la de Fig. B.5 en la que se podrán cambiar parámetros como el nombre, la dirección MAC o la dirección IP.

Los ordenadores tienen una función especial que es la de crear y enviar los paquetes de red. Para ello, en el menú emergente que aparece al hacer click derecho en el objeto, hacemos click en la entrada de 'Ping'. Para que el paquete llegue al otro computador, ambos deben tener una dirección IP, y estar conectados a la misma red. Se introduce la dirección IP del dispositivo y se pulsa en el botón de 'Ping!' (Ver Fig. B.2 y Fig. B.3). A continuación veremos el paquete de red buscando su objetivo, la primera vez no irá directamente, ya que los Switches están aún aprendiendo el camino, pero el paquete de vuelta y todos los siguientes paquetes seguirán la misma ruta (Ver Fig. B.6). El ordenador crea un paquete de red usando los protocolos de Ethernet (IEEE 802.11), TCP, IPv4 e ICMP. La función que se encarga de esto es `create`.

Los Switches se encargan de redireccionar los paquetes de red. La primera vez que les llega un paquete, al no saber la ubicación física del destino, siguen este algoritmo:

```

1268         #Si macd en conn, enviarle el paquete
1269         #Si existe una tabla de enrutamiento que contiene una ruta para macd, enviar por ahi
1270         #Si no, enviar al siguiente, y así
1271         print(">MAAAC:",int(macd,2), "DIIIC:")
1272         if int(macd,2) in dic and ttl > 0:
1273             pck.animate(self, dic[int(macd,2)])
1274
1275         elif int(macd,2) in [x[0] for x in self.table] and ttl >= 0:
1276             for x in self.table:
1277                 if x[0] == int(macd,2):
1278                     pck.animate(self, self.pdic[x[1]])
1279
1280         elif "Switch" in [x.objectype for x in self.connections] and ttl >= 0:
1281             print("Ahora lo enviamos al siguiente router")
1282             print(int(macd,2), dic)
1283             tmp1st = self.connections[:] #Crea una nueva copia de la lista
1284             print(tmp1st)
1285             for i in tmp1st:
1286                 if int(macs,2) == int(i.macdir):
1287                     print("REMOVING", i)
1288                     tmp1st.remove(i)
1289             try:
1290                 tmp1st.remove(*[x for x in tmp1st if x.objectype == "Computer"])
1291             except TypeError:
1292                 pass

```

```

1293         print("Tplst:", tplst)
1294         obj = choice(tmplst)
1295         print("Sending to:", obj)
1296         pck.animate(self, obj)

```

Este algoritmo esta basado en el que usan los conmutadores reales y, traducido a lenguaje humano, vendría a ser:

Si la dirección MAC de destino del paquete recibido se encuentra directamente conectado al Switch y el TTL del paquete es mayor que cero:

Enviar el paquete a ese dispositivo.

Al no cumplirse la condición anterior, si el paquete se encuentra en mi tabla de enrutamiento y el TTL del paquete es mayor que cero:

Enviamos el paquete por el puerto al que está asignada la dirección MAC en la tabla.

Al no cumplirse las condiciones anteriores, si hay un Switch en mis conexiones y el TTL del paquete es mayor a 0:

Enviar el paquete a uno de los Switches de forma aleatoria.

Cuando recibe un paquete, también añade a la *Routing Table* o tabla de enrutación una entrada con la dirección MAC del remitente del paquete y el puerto por el que ha llegado, así cuando le llegue un paquete el router conocerá el puerto por el que enviarlo.

```

1231         for tab in self.table:
1232             if tab[2] <= time.time():
1233                 print("Ha llegado tu hora")
1234                 self.table.remove(tab)
1235                 self.wtable.remove(tab)
1236             if tab[0] == int(macd,2):
1237                 print("TAB[0] == mcd")
1238                 tab[2] = int(time.time()+self.timeout)
1239                 for row in self.wtable.store:
1240                     print(row[0], tab[0])
1241                     if int(row[0].replace(":", ""),16) == tab[0]:
1242                         row[3] = int(time.time()+self.timeout)
1243             if int(macs,2) not in [x[0] for x in self.table]:
1244                 tmp = [int(macs,2), port, int(time.time()+self.timeout)]
1245                 self.table.append(tmp)
1246                 tmp = [readmac, port, int(time.time()+self.timeout)]
1247                 self.wtable.append(tmp)

```

Este es el código que cumple esta función. Cada elemento en la tabla tiene un tiempo establecido en el que caduca la entrada. Lo que hace esta parte del código es comprobar si este tiempo ha caducado, actualizar la fecha de caducidad si la dirección MAC ya está en la tabla o añadirlo de nuevo en la tabla si la dirección no está.

3.5 Versión actual del programa (0.2.3-alpha)

En la versión 0.1 se introdujo toda la interfaz, las conexiones, los dispositivos... Pero aún no se podían enviar ni recibir paquetes de red. En la versión 0.2 se introdujo esta posibilidad, junto a otras cosas como el enrutamiento de paquetes. El programa es considerado una versión *alpha*, ya que aún está en desarrollo y no es un programa terminado.

El programa te permite, por el momento, hacer una simulación de red simple. Se podría decir que es una base sobre la que se pueden ir añadiendo más funcionalidades, como el soporte para otros protocolos, o un modo ‘explicatorio’ que enseñe a los alumnos lo que está pasando en la red. En la versión 0.2.3-alpha del programa sólo se ha introducido el “Ping”, es decir, la posibilidad de enviar un paquete de prueba a otro dispositivo de la misma red. También se han introducido algunos cambios en la interfaz, uno de ellos, bastante útil para el aprendizaje: los cuadros de texto en los que se introducen direcciones IP, cambian de color entre rojo, naranja o verde, dependiendo si la IP introducida no es válida, está incompleta o es válida, respectivamente.

3.6 Desarrollo del proyecto

En cuanto al código, a pesar de la gran extensión del programa, han sido escritas muchas más líneas, que han sido en algún momento eliminadas o reemplazadas. El desarrollo del proyecto puede dividirse en 4 fases a lo largo de 3 trimestres.

En la primera fase, de Noviembre a Febrero/Marzo he ido aprendiendo sobre todo de Gtk+, la librería para la interfaz del programa. Al empezar el proyecto mis conocimientos sobre esta librería eran nulos; y sobre Python, el lenguaje de programación, eran demasiado básicos. También aprendí bastante sobre redes informáticas.

En la segunda fase, se fue desarrollando la “base” del programa, transcurre de Febrero-Marzo a finales de curso. La interfaz, las ideas, las conexiones de los cables... Se construye la versión 0.1, como menciono en 3.5. El programa contaba con unas 700 líneas en `Main.py`

En la tercera fase se desarrolla la gran parte del programa, aquí es cuando llega a las 2000 líneas, sin mencionar los pequeños módulos y otros archivos. Transcurre en verano, entre Junio y mediados de Agosto. Con una media de 200-300 líneas semanales y picos de hasta mil líneas entre el 7 y el 14 de Agosto, ha sido posible cumplir el objetivo de crear un pequeño simulador

de redes. Es el desarrollo de la versión 0.2-alpha, ya que el programa sigue en desarrollo de posteriores versiones.

La cuarta fase transcurre solapada con la tercera, comienza en Julio y acaba el día 4 de enero de 2017, con la entrega de la memoria del proyecto. Es la fase en la que se desarrolla esta memoria.

He notado bastante la adquisición de experiencia, ya que tardé prácticamente 5-6 meses en hacer las primeras 500 líneas; pero en verano, conforme iba programando más, conseguí llegar a hacer más de 1000 líneas en una semana. También, al leer el código antiguo se notan bastantes errores debidos a la falta de experiencia, que tiene que ser corregidos si aún no se ha hecho.

3.6.1 Obstáculos en el desarrollo del proyecto

Durante el desarrollo del proyecto han surgido bastantes trabas y contratiempos, que he conseguido solucionar. Muchos de ellos surgen por la falta del gran conocimiento técnico necesario para la creación de un software tan específico, han sido muchas horas de mirar la documentación de las librerías [6], y pedir ayuda por foros para intentar solucionar dudas y bugs.

En algunas ocasiones no han sido errores, sino falta de conocimiento para el desarrollo de determinadas funciones lo que ha creado pausas de hasta dos semanas en la acción de escribir el programa. Gracias a comunidades como *stackoverflow* he conseguido solucionar muchas de las dudas y errores básicos del programa.

Otro inconveniente ha sido el tiempo, que no me ha dejado implementar funciones útiles, como un visor de paquetes de red, más protocolos, o interconexión de redes.

3.7 Conclusión

Lo más difícil fue empezar. El tratar de aprender tanta información de golpe de forma autodidacta. Aunque ya supiese un poco sobre programación en Python, no tenía casi experiencia, aprender a usar la librería de Gtk+, aprender sobre redes, aprender sobre un uso más extenso de GNU/Linux, aprender sobre L^AT_EX, etc. fue bastante cansado. pero eso es lo mejor, todo lo que he aprendido y, sobre todo, la experiencia que he adquirido en el campo de la programación.

A la hora de programar, al principio el ritmo era muy lento, de unas 200 líneas al mes, con pausas de semanas para solucionar problemas y errores. Poco a poco se fue acelerando hasta llegar a finales de Julio, donde hacía más de 100 líneas diarias.

Pese a que es verdad que falta incluir más protocolos y algunas funcionalidades bastante básicas (como mover objetos), estoy bastante satisfecho con la versión actual del programa, que se ha realizado con bastante poco tiempo, ya que tiene las bases, y creo que añadir un nuevo protocolo, o una nueva funcionalidad no serían más que unas horas delante de la pantalla y el teclado del ordenador.

Resumiendo:

- Se ha creado un simulador de redes escrito en Python que demuestra el uso de Ping's y enrutamiento de paquetes
- Con el programa actual, es fácil añadir nuevos paquetes, dispositivos o funcionalidades, entre otras cosas
- Se ha desarrollado íntegramente con software libre
- Tiene uso didáctico en 4º de la ESO, 1º y 2º de Bachillerato, pues se incluye 'Redes informáticas' en el temario de Informática y TICO, además de 'Programación'
- Los alumnos pueden modificar y ampliar el programa, para toda la comunidad educativa, en GitHub

Bibliografía

- [1] Cisco. (2015). Cisco visual networking index: Forecast and methodology, dirección: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html.
- [2] Free Software Foundation. (2013). Filosofía del proyecto GNU, dirección: <https://www.gnu.org/philosophy/philosophy.html>.
- [3] R. Stallman. (2014). Charla: Free software, free society: R. Stallman at TEDxGeneva, dirección: https://www.youtube.com/watch?v=Ag1AKI1_2GM.
- [4] — —, (2013). Conferencia sobre software libre en la universidad de Jaume I, dirección: https://www.youtube.com/watch?v=5t_EcPTEzh4.
- [5] BICSI, *Network Design Basics for Cabling Professionals*. 2002.
- [6] C. Reiter (lazka en GitHub). (2016). Python GObject Introspection API Reference, dirección: <https://lazka.github.io/pgi-docs/>.
- [7] Python Software Foundation. (2016). What is Python? executive summary, dirección: <https://www.python.org/doc/essays/blurb/>.
- [8] Real Academia Española, *Diccionario de la lengua española*, ed. XXIII. 2014.
- [9] R. Braden, *Request for Comments 1122*, 1989.
- [10] Alumnado de la asignatura de Software Libre del Máster en Sistemas Telemáticos e Informáticos de la Universidad Rey Juan Carlos. (2013). Traducción de la licencia GPLv3 al español, dirección: https://lslspanish.github.io/translation_GPLv3_to_spanish/.
- [11] University of Cambridge Computer Laboratory. (2001). A brief informal history of the computer laboratory, dirección: <https://www.cl.cam.ac.uk/events/EDSAC99/history.html>.
- [12] Microsoft Developer Network. (2015). Serialización, dirección: <https://msdn.microsoft.com/es-es/library/ms233843.aspx?f=255&MSPPErrors=-2147217396>.
- [13] Wikipedia. (2016). Local area network: History, dirección: https://en.wikipedia.org/wiki/Local_area_network#History.
- [14] All About Circuits. (2016). Introduction to boolean algebra, dirección: <http://www.allaboutcircuits.com/textbook/digital/chpt-7/introduction-boolean-algebra/>.

Glosario y acrónimos

ADSL *Asymmetric Digital Subscriber Line*, Línea de Abonado Digital Asimétrica

Bit *Binary digit*, o dígito binario. Cada dígito del sistema de numeración binario

Botnet Grupo de ordenadores coordinados conectados a un maestro mediante un virus. Gracias a este virus se pueden realizar tareas masivas como el envío de SPAM o ataques DDoS

Bug Cualquier tipo de error en el código de un programa informático. Por ejemplo, que un botón no realice la acción que debería de hacer.

Caché Almacenamiento temporal de datos con el objetivo de reducir el retardo, la carga de los servidores y el ancho de banda consumido

Capas de abstracción Método de ocultar detalles de implementación de un set de funcionalidades

Conmutación de paquetes Método para enviar datos por una red de computadoras. Se divide el paquete en dos partes, una con información de control que leen los nodos para enviar el paquete a su destino y los datos a enviar

Datos Secuencia binaria de unos y ceros que contiene información codificada

Dependencia De un programa, otro tipo de software necesario para que éste funcione

FSF *Free Software Foundation*, Fundación del Software Libre

FTTH *Fiber To The Home* [Fibra hasta el hogar]

FTTx *Fiber to the X*

GNU *GNU's Not Unix*, GNU no es Unix

GUI Interfaz Gráfica de Usuario, *Graphic User Interface*)

Hardware Conjunto de elementos físicos o materiales que constituyen un sistema informático.

IDE Entorno de Desarrollo Integrado, *Integrated Development Enviroment*

IEEE Instituto de Ingeniería Eléctrica y Electrónica

International Organization for Standardization Organización Internacional de Normalización. Compuesta de varias organizaciones nacionales se encarga de la creación de estándares internacionales desde 1947.

ISO *International Organization for Standardization*

LAN *Local Area Network* [Red de Área Local]

Latencia También conocido como *lag*, es la suma de los retardos producido en el envío o la recepción de datos.

Librería En informática, una librería o biblioteca es un conjunto de recursos y funciones diseñadas para ser usadas por otros programas. Incluyen plantillas, funciones y clases, subrutinas, código escrito, variables predefinidas...

MAC *Media Access Control*, Control de Acceso al Medio

OSI *Open Systems Interconnection* (Interconexión de Sistemas Abiertos)

Ping Es un programa que envía o responde un paquete ICMP. Sirve para determinar si

dos dispositivos están conectados en una red y, por lo tanto, comprobar la conexión.

POP3 *Post Office Protocol*, Protocolo de Oficina Postal

Programación imperativa Las órdenes del programa cambian el estado de este mismo. Por ejemplo, una variable no tiene por qué ser declarada con antelación y su valor es modificable. Es la que usa el código máquina de los ordenadores

Repositorio Servidor donde se alojan ficheros o archivos para su descarga

Serialización La serialización es el proceso de convertir un objeto en una secuencia de bytes para almacenar el objeto o transmitirlo

a memoria, una base de datos, o en un archivo. Su propósito principal es guardar el estado de un objeto para poder crearlo de nuevo cuando se necesita. El proceso inverso se denomina deserialización. [12]

Topología “Rama de las matemáticas que trata especialmente de la continuidad y de otros conceptos más generales originados de ella, como las propiedades de las figuras con independencia de su tamaño o forma.” [8][Topología]

Topología de red Configuración espacial o física de la red. (Ver 2.2 pág.7)

URL *Uniform Resource Identifier*, Identificador de Recursos Uniforme

Apéndice A

Unidades de transferencia de datos

Cantidad de datos transferidos por unidad de tiempo. La unidad de tiempo es el segundo y la cantidad de datos puede ser medida en *bits* (bitrate), caracteres/símbolos (*baudrate*) o bytes (8 bits), en ocasiones también se utilizan *nibbles* (4 bits). Para expresar esta velocidad, se suelen usar múltiplos, que pueden ser en base binaria (Sistema del IEEE) o decimal (Sistema Internacional).

Se usa la “b” para designar los bits, y “B” para los Bytes. Después, se usan los prefijos del sistema internacional cuando es en base decimal, y los prefijos del SI cambiando la segunda sílaba por “bi” (e.g. kilobit / kibibit, kbit/s / Kibit/s) cuando se trata de múltiplos binarios.

Tabla de múltiplos

Unidad	Símbolo	Equivalencia
Kilobit/s	kbit/s o kb/s	1000 bit/s
Megabit/s	Mbit/s o Mb/s	10^6 bit/s o 10^3 kbit/s
Gigabit/s	Gbit/s o Gb/s	10^9 bit/s o 10^3 Mb/s
Terabit/s	Tbit/s o TB/s	10^{12} bit/s o 10^3 Gb/s
Kibibit/s	Kibit/s	2^{10} bit/s o 1024 bit/s
Mebibit/s	Mibit/s	2^{20} bit/s o 1024 Kibit/s
Gibibit/s	Gibit/s	2^{30} bit/s o 1024 Mibit/s
Tebibit/s	Tibit/s	2^{40} bit/s o 1024 Gibit/s
Byte/s	Byte/s	8 bit/s
Kilobyte/s	kB/s	1000 Byte/s o 8000 bits/s
Megabyte/s	MB/s	10^6 Byte/s o 1000 kB/s
Gigabyte/s	GB/s	10^9 Byte/s o 1000 MB/s
Terabyte/s	TB/s	10^{12} Byte/s o 1000 GB/s
Kibibyte/s	KiB/s	1024 Byte/s
Mebibyte/s	MiB/s	2^{20} Byte/s
Gibibyte/s	GiB/s	2^{30} Byte/s
Tebibyte/s	TiB/s	2^{40} Byte/s

Apéndice B

Capturas de pantalla del programa

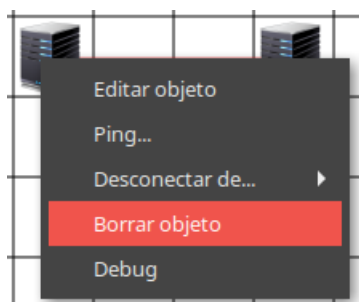


Figura B.1: Captura: Click derecho en un computador

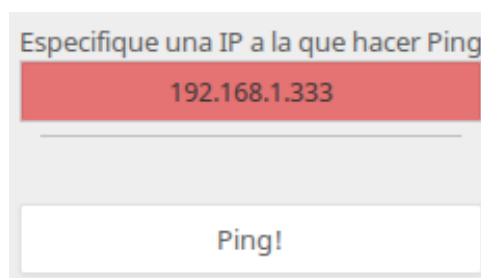


Figura B.2: Captura: Ventana para enviar ping. Está en rojo porque la IP introducida no es válida.

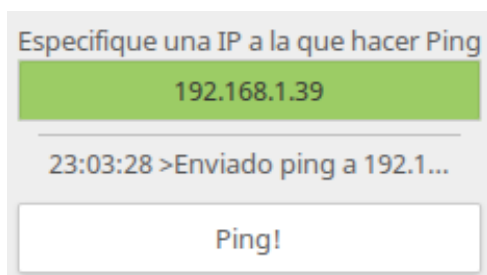


Figura B.3: Captura: Igual que B.2, pero con una IP válida.

MAC	Puerto	TTL (s)
EA:6F:0B:4F:F0:EE	2	284
EE:99:CC:0C:4A:4B	3	241
E4:F2:B9:A2:8C:F9	3	272
ED:9F:48:97:54:FE	1	274
D5:15:5C:02:DD:04	3	277
F9:FC:E6:21:E9:6F	3	280
C2:70:AA:11:08:CF	3	281

⌂ Cerrar

Figura B.4: Captura: Ventana con la tabla que posee el Switch.

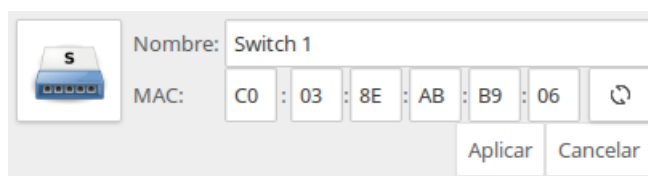


Figura B.5: Captura: Ventana de edición de propiedades de objeto.

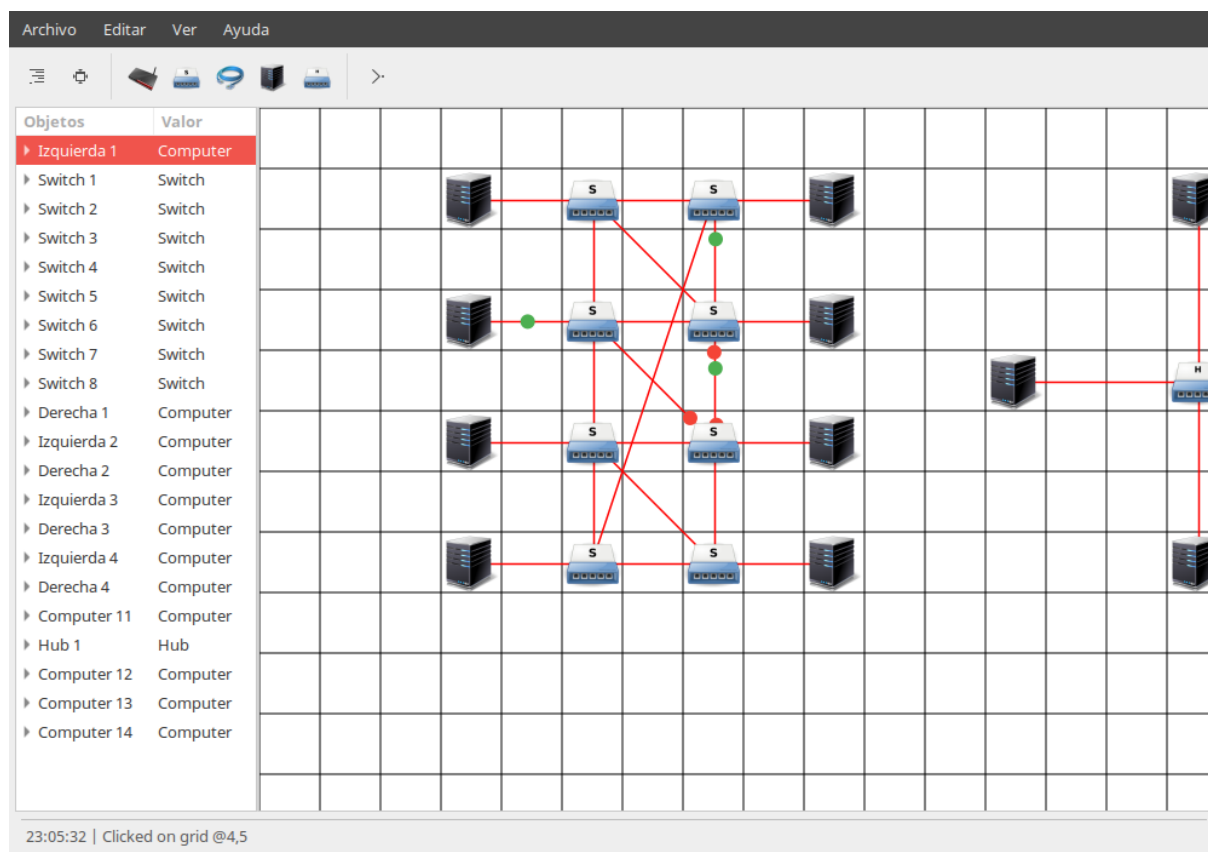


Figura B.6: Captura: Paquetes viajando por una red de ejemplo.

Apéndice C

Licencia Pública General GNU

Es la licencia más usada en el desarrollo de software. Permite las cuatro libertades del software libre y fue creada por la Free Software Foundation. La versión más reciente, la GPLv3, ha sido publicada el 29 de junio de 2007. Al distribuir el programa, debe distribuirse también una copia de la licencia que usa.

Este documento está licenciado con licencia Creative Commons Reconocimiento-CompartirIgual 4.0 Internacional, compatible con la GNU GPL.

InvProy usa la licencia GPLv3, aquí una traducción no oficial[10]:

```
1 ***** LICENCIA PÚBLICA GENERAL DE GNU *****
2 **** Versión 3, 29 de junio de 2007 ****
3 Esta es una traducción no oficial al español de la GNU General Public License.
4 No ha sido publicada por la Free Software Foundation, y no establece legalmente
5 las condiciones de distribución para el software que usa la GNU GPL estas
6 condiciones se establecen solamente por el texto original, en inglés, de la GNU
7 GPL. Sin embargo, esperamos que esta traducción ayude a los hispanohablantes a
8 entender mejor la GNU GPL.
9
10 This is an unofficial translation of the GNU General Public License into
11 Spanish. It was not published by the Free Software Foundation, and does not
12 legally state the distribution terms for software that uses the GNU GPL only
13 the original English text of the GNU GPL does that. However, we hope that this
14 translation will help Spanish speakers understand the GNU GPL better.
15
16 Copyright © 2007 Free Software Foundation, Inc. http://fsf.org/
17
18
19 Se permite la copia y distribución de copias literales de este documento, pero
20 no se permite su modificación.
21
22   Preámbulo: La Licencia Pública General de GNU es una licencia libre,
23   bajo "copyleft", para software y otro tipo de obras.
24   Las licencias para la mayoría del software y otras obras de carácter
25   práctico están diseñadas para privarle de la libertad de compartir y
26   modificar las obras. Por el contrario, la Licencia Pública General de
27   GNU pretende garantizar su libertad de compartir y modificar todas
28   las versiones de un programa para cerciorar que permanece como
29   software libre para todos sus usuarios. Nosotros, la Free Software
30   Foundation, usamos la Licencia Pública General de GNU para la mayoría
31   de nuestro software; la cual se aplica también a cualquier otra obra
32   publicada de esta forma por parte de sus autores. Usted también puede
33   aplicarla a sus programas.
34
35   Cuando hablamos de software libre (free software), nos referimos a
36   libertad, no a precio. Nuestras Licencias Públicas Generales están
37   diseñadas para garantizar su libertad de distribuir copias de
38   software libre (y cobrar por ellas si lo desea), recibir el código
39   fuente o poder obtenerlo si quiere, modificar el software o usar
40   fragmentos de él en sus nuevos programas, y que sepa que puede hacer
41   esas cosas.
42
43   Para proteger sus derechos, necesitamos impedir que otros le
44   denieguen esos derechos o que le pidan que renuncie a ellos. Por
45   ello, tiene ciertas responsabilidades si distribuye copias del
46   software, o si lo modifica: la responsabilidad de respetar la
```


libertad de otros.

Por ejemplo, si distribuye copias de un programa, bien sea gratis o por una tasa, debe transferirles a los que lo reciban las mismas libertades que usted recibió. Debe asegurarse que ellos, también, reciben o pueden obtener el código fuente. Y debe mostrarles estos términos para que ellos puedan conocer sus derechos.

Los desarrolladores que usan la GNU GPL protegen tus derechos con dos pasos: (1) haciendo valer el derecho de propiedad intelectual en el software, y (2) ofreciéndole esta Licencia que le da el permiso legal para copiarlo, distribuirlo y/o modificarlo.

Para la protección de autores y desarrolladores, la GPL explica claramente que no hay garantía para este software libre. Por el bien tanto de usuarios como de autores, la GPL requiere que las versiones modificadas sean marcadas como con cambios, de forma que sus problemas no puedan ser atribuidos de forma errónea a autores de versiones previas.

Algunos dispositivos están diseñados para denegar a los usuarios el acceso para instalar o ejecutar versiones modificadas del software en su interior, a pesar de que el fabricante puede hacerlo. Esto es fundamentalmente incompatible con el objetivo de proteger la libertad de los usuarios de modificar el software. El modelo sistemático de este abuso ocurre en el ámbito de los productos de uso personal, lo cual es precisamente donde es más inaceptable. Por consiguiente, hemos diseñado esta versión de la GPL para prohibir la práctica de estos productos. Si estos problemas surgen de forma substancial en otro dominios, estamos preparados para extender esta disposición a esos dominios en futuras versiones de la GPL, así como sea necesario para proteger la libertad de los usuarios.

Por último, todo programa es amenazado constantemente por las patentes de software. Los Estados no deberían permitir patentes que restrinjan el desarrollo y el uso de software en ordenadores de propósito general, pero en aquellos que lo hacen, deseamos evitar el peligro particular de que las patentes aplicadas a un programa libre podrían convertirlo de forma efectiva en propietario. Para prevenir esto, la GPL garantiza que las patentes no pueden ser utilizadas para hacer que el programa no sea libre.

Los términos exactos y las condiciones para la copia, distribución y modificación se exponen a continuación.

Términos y Condiciones

1. Definiciones.

“Esta Licencia” se refiere a la versión 3 de la Licencia Pública General de GNU.

“Derechos de Autor (“Copyright”)” también incluye a las leyes similares a la de derechos de autor (“copyright”) que se apliquen a otro tipo de obras, tales como las máscaras usadas en la fabricación de semiconductores.

“El Programa” se refiere a cualquier obra con derechos de autor (“copyright”) bajo esta Licencia. Cada licenciataria es tratado como “usted”. Los “Licenciarios” y los “destinatarios” pueden ser individuos u organizaciones.

“Modificar” una obra quiere decir copiar de ella o adaptar parte o la totalidad de la obra de una forma que se requieran permisos de derechos de autor (“copyright”), distintos de los de hacer una copia exacta. La obra resultante es llamada “versión modificada” de la obra previa o una obra “basada en” la obra previa.

Una “obra amparada” significa o el Programa sin modificar o una obra basada en el Programa.

“Difundir” una obra significa hacer cualquier cosa con ella que, sin permiso, le haría responsable de forma directa o indirecta de infringir la ley correspondiente de derechos de autor (“copyright”), excepto ejecutarla en un ordenador o modificar una copia privada. La difusión incluye copiar, la distribución (con o sin modificación), hacerla disponible para el público, y en algunos países también otras actividades.

“Transmitir” una obra quiere decir cualquier tipo de difusión que permita a otras partes hacer o recibir copias. La mera interacción con un usuario a través de una red informática, sin la transferencia de una copia, no es transmitir.

Una interfaz interactiva de usuario muestra “Avisos Legales Apropriados” en la medida que incluye una característica visible práctica y destacable

que (1) muestra un aviso apropiado de derechos de autor ("copyright"), e (2) informa al usuario de que no hay garantía para la obra (excepto las garantías proporcionadas), que los licenciarios pueden transmitir la obra bajo esta licencia, y cómo ver una copia de esta Licencia. Si la interfaz presenta una lista de comandos de usuario u opciones, como un menú, un elemento destacado en la lista satisface este criterio.

2. Código Fuente.

El "código fuente" de una obra significa la forma preferida de trabajo para hacerle modificaciones. "Código objeto" es cualquier forma no-fuente de una obra.

Una "Interfaz Estándar" significa una interfaz que es un estándar oficial definido por un cuerpo de estándares reconocido o, en el caso de interfaces especificadas para un lenguaje de programación en particular, una que es extensamente utilizada entre los desarrolladores que trabajan en ese lenguaje.

Las "Bibliotecas del Sistema" de una obra ejecutable incluyen cualquier cosa, diferente de la obra como un todo, que (a) están incluidas en la forma normal de paquetizado de un Componente Importante, y (b) sirve solo para habilitar el uso de la obra con ese Componente Importante, o para implementar una Interfaz Estándar para la cual la implementación está disponible para el público en forma de código fuente. Un "Componente Importante", en este contexto, significa un componente esencial importante (kernel, sistema de ventanas, etcétera) del sistema operativo en concreto (si hubiese) en el cual el ejecutable funciona, o un compilador utilizado para producir la obra, o un intérprete de código objeto utilizado para hacerlo funcionar.

La "Fuente Correspondiente" de una obra en forma de código objeto significa todo el código fuente necesario para generar, instalar, y (para una obra ejecutable) hacer funcionar el código objeto y modificar la obra, incluyendo scripts para controlar dichas actividades. Sin embargo, ello no incluye la obra de las Bibliotecas del Sistema, o herramientas de propósito general o programas de libre disponibilidad general los cuales son usados sin modificaciones para la realización de dichas actividades, pero que no son parte de la obra. Por ejemplo, la Fuente Correspondiente incluye ficheros de definición de interfaces asociados a los ficheros fuente para la obra, y el código fuente para bibliotecas compartidas y subprogramas enlazados dinámicamente para los que la obra está específicamente diseñado para requerir, tales como comunicación de datos intrínseca o flujo de control entre aquellos subprogramas y otras partes de la obra.

La Fuente Correspondiente es necesario que no incluya nada que los usuarios puedan regenerar automáticamente desde otras partes de la Fuente Correspondiente.

La Fuente Correspondiente de una obra en forma de código fuente es la obra en sí.

3. Permisos básicos.

Todos los derechos concedidos bajo esta Licencia se conceden durante la duración de los derechos de autor ("copyright") del Programa, y son irrevocables siempre que se cumplan las condiciones establecidas. Esta Licencia afirma explícitamente su ilimitado permiso para ejecutar el Programa sin modificar. La salida de la ejecución de una obra amparada está amparada por esta Licencia solo si la salida, dado su contenido, constituye una obra amparada. Esta Licencia reconoce sus derechos de uso razonable u otro equivalente, según lo establecido por la ley de derechos de autor ("copyright").

Usted podrá realizar, ejecutar y difundir obras amparadas que usted no transmita, sin condición alguna, siempre y cuando no tenga otra licencia vigente. Podrá distribuir obras amparadas a terceros con el único propósito de que ellos hagan modificaciones exclusivamente para usted, o proporcionarle ayuda para ejecutar estas obras, siempre y cuando cumpla con los términos de esta Licencia en la transmisión de todo el material del cual usted no controle los derechos de autor ("copyright"). Aquellos que realicen o ejecuten las obras amparadas por usted, deben hacerlo exclusivamente en su nombre, bajo su dirección y control, en los términos que le prohíban realizar ninguna copia de su trabajo con derechos de autor ("copyright") fuera de su relación con usted.

La transmisión bajo otras circunstancias se permite únicamente bajo las condiciones expuestas a continuación. No está permitido sublicenciar, la sección 10 hace que sea innecesario.

4. Protección de los Derechos Legales de los Usuarios frente a la Ley

Antievasión.

Ninguna obra amparada debe considerarse parte de una medida tecnológica efectiva, a tenor de lo establecido en cualquier ley aplicable que cumpla las obligaciones expresas en el artículo 11 del tratado de derechos de autor ("copyright") de WIPO adoptado el 20 de diciembre de 1996, o leyes similares que prohíban o restrinjan la evasión de tales medidas.

Cuando transmita una obra amparada, renuncia a cualquier poder legal para prohibir la evasión de medidas tecnológicas mientras tales evasiones se realicen en ejercicio de derechos amparados por esta Licencia respecto a la obra amparada; además, usted renunciará a cualquier intención de limitar el uso o modificación del trabajo con el objetivo de imponer, contra el trabajo de los usuarios, sus derechos legales o los de terceros para prohibir la evasión de medidas tecnológicas.

5. Transmisión de copias literales.

Usted podrá distribuir copias literales del código fuente del Programa tal cual lo ha recibido, por cualquier medio, siempre que publique visible y apropiadamente en cada copia el correspondiente aviso de derechos de autor ("copyright"); mantenga intactos todos los avisos que establezcan que esta Licencia y cualquier cláusula no-permisiva añadida acorde con la cláusula 7 son aplicables al código; mantenga intactos todos los avisos de ausencia de garantía; y proporcione a todos los destinatarios una copia de esta Licencia junto con el Programa. Usted podrá cobrar cualquier importe o no cobrar nada por cada copia que distribuya, y podrá ofrecer soporte o protección de garantía mediante un pago.

6. Transmisión de Versiones Modificadas de la Fuente.

Usted puede transmitir una obra basada en el Programa, o las modificaciones para generarla a partir del Programa, en la forma de código fuente bajo los términos de la sección 4, suponiendo que además cumpla las siguientes condiciones:

- a. La obra debe incluir avisos destacados indicando que usted la ha modificado y dando una fecha pertinente.
- b. La obra debe incluir avisos destacados indicando que está liberada bajo esta Licencia y cualquier otra condición añadida bajo la sección 7. Este requerimiento modifica los requerimientos de la sección 4 de "mantener intactos todos los avisos".
- c. Usted debe licenciar la obra entera, como una unidad, bajo esta Licencia para cualquier persona que esté en posesión de una copia. Esta Licencia se aplicará por consiguiente, junto con cualquier término aplicable adicional de la sección 7, a la totalidad de la obra, y a todos sus componentes, independientemente de como estén empaquetados. Esta Licencia no da permiso para licenciar la obra de otra forma, pero no invalida esos permisos si usted los ha recibido de forma separada.
- d. Si la obra tiene interfaces de usuario interactivas, cada una debe mostrar los Avisos Legales Apropriados; sin embargo, si el Programa tiene interfaces interactivas que no muestren los Avisos Legales Apropriados, tampoco es necesario que su obra lo haga.

Una recopilación de una obra amparada con otras obras separadas e independientes, que no son por su naturaleza extensiones de la obra amparada, y que no se combinan con ella con el fin de formar un programa más grande, en o sobre un volumen de un medio de almacenamiento o distribución, es llamado un "agregado" si la recopilación y su resultante derechos de autor ("copyright") no son usados para limitar el acceso o los derechos legales de los usuarios de la recopilación más allá de lo que las obras individuales permitan. La inclusión de una obra amparada en un agregado no provoca que esta Licencia se aplique a los otros componentes del agregado.

7. Transmisión en Forma de No-Fuente.

Usted puede transmitir una obra amparada en forma de código objeto bajo los términos de las secciones 4 y 5, siempre que también transmita la Fuente Correspondiente legible por una máquina bajo los términos de esta Licencia, de una de las siguientes formas:

- a. Transmitir el código objeto en, o embebido en, un producto físico (incluyendo medios de distribución físicos), acompañado de la Fuente Correspondiente en un medio físico duradero habitual para el intercambio de software.
- b. Transmitir el código objeto en, o embebido en, un producto físico (incluyendo medios de distribución físicos), acompañado de un ofrecimiento escrito, válido durante al menos tres años y válido

mientras usted ofrezca recambios o soporte para ese modelo de producto, de dar a cualquiera que posea el código objeto o (1) una copia de la Fuente Correspondiente de todo el software en el producto amparado por esta Licencia, en un medio físico duradero habitual para el intercambio de software, por un precio no más elevado que el coste razonable de la realización física de la transmisión de la fuente, o (2) acceso para copiar la Fuente Correspondiente de un servidor de red sin costo alguno.

- c. Transmitir copias individuales del código objeto con una copia del ofrecimiento escrito de proveer la Fuente Correspondiente. Esta alternativa está permitida solo ocasionalmente sin fines comerciales, y solo si usted ha recibido el código objeto con ese ofrecimiento, de acuerdo con la subsección 6b.
- d. Transmitir el código objeto ofreciendo acceso desde un lugar determinado (gratuitamente o mediante pago), y ofrecer acceso equivalente a la Fuente Correspondiente de la misma manera en el mismo lugar sin cargo adicional. No es necesario exigir a los destinatarios que copien la Fuente Correspondiente junto con el código objeto. Si el lugar para copiar el código objeto es un servidor de red, la Fuente Correspondiente puede estar en un servidor diferente (gestionado por usted o un tercero) que soporte facilidades de copia equivalentes, siempre que mantenga instrucciones claras junto al código objeto especificando dónde encontrar la Fuente Correspondiente. Independientemente de qué servidor albergue la Fuente Correspondiente, usted seguirá estando obligado a asegurar que está disponible durante el tiempo que sea necesario para satisfacer estos requisitos.
- e. Transmitir el código objeto usando una transmisión peer-to-peer, siempre que informe a los otros usuarios donde se ofrece el código objeto y la Fuente Correspondiente de la obra al público general de forma gratuita bajo la subsección 6d.

Una porción separable del código objeto, cuyo código fuente está excluido de la Fuente Correspondiente, como una Biblioteca del Sistema, no necesita ser incluida en la distribución del código objeto de la obra.

Un “Producto de Usuario” es o (1) un “producto de consumo”, lo que significa cualquier propiedad tangible personal que es usada habitualmente con fines personales, familiares o domésticos, o (2) cualquier cosa diseñada o vendida para ser incorporada en una vivienda. A la hora de determinar cuando un producto es un producto de consumo, los casos dudosos serán resueltos en favor de la cobertura. Para un producto concreto recibido por un usuario concreto, “uso habitual” se refiere a un uso típico y común de esa clase de producto, sin tener en cuenta el estado del usuario concreto o la forma en la que el usuario concreto realmente use, o espera o se espera que use, el producto. Un producto es un producto de consumo independientemente de si el producto tiene usos esencialmente comerciales, industriales o no comerciales, a menos que dicho uso constituya el único modo de uso significativo del producto. La “Información de Instalación” de un Producto de Usuario quiere decir cualquier método, procedimiento, clave de autorización, u otra información requerida para instalar y ejecutar versiones modificadas de la obra amparada en ese Producto de Usuario a partir de una versión modificada de su Fuente Correspondiente. La información debe ser suficiente para garantizar que el funcionamiento continuado del código fuente modificado no es prevenido o interferido por el simple hecho de que ha sido modificado.

Si usted transmite una obra en código objeto bajo esta sección en, o con, o específicamente para usar en, un Producto de Usuario, y la transmisión tiene lugar como parte de una transacción en la cual el derecho de posesión y uso de un Producto de Usuario es transferido a un destinatario en perpetuidad o por un periodo establecido (independientemente de cómo se caracterice la operación), la Fuente Correspondiente transmitida bajo esta sección debe estar acompañada de la Información de Instalación. Pero este requisito no se aplica si ni usted ni ningún tercero tiene la capacidad de instalar código objeto modificado en el Producto de Usuario (por ejemplo, la obra ha sido instalada en la ROM).

El requisito de proveer de la Información de Instalación no incluye el requisito de continuar proporcionando asistencia, garantía, o actualizaciones para una obra que ha sido modificada o instalada por el destinatario, o para un Producto de Usuario en el cual ha sido modificada o instalada. El acceso a una red puede ser denegado cuando la

modificación en sí afecta materialmente y adversamente el funcionamiento de la red o viola las reglas y protocolos de comunicación de la red. La Fuente Correspondiente transmitida, y la Información de Instalación proporcionada, de acuerdo con esta sección debe estar en un formato que sea documentado públicamente (y con una implementación disponible para el público en formato de código fuente), y no deben necesitar contraseñas o claves particulares para la extracción, lectura o copia.

8. Términos adicionales.

Los “Permisos adicionales” son términos que se añaden a los términos de esta Licencia haciendo excepciones de una o más de una de sus condiciones. Los permisos adicionales que son aplicables al Programa entero deberán ser tratados como si estuvieran incluidos en esta Licencia, en la medida bajo la ley aplicable. Si los permisos adicionales solo son aplicables a parte del Programa, esa parte debe ser usada separadamente bajo esos permisos, pero el Programa completo queda bajo la autoridad de esta Licencia sin considerar los permisos adicionales. Cuando se transmite una copia de una obra derivada, se puede opcionalmente quitar cualesquiera permisos adicionales de esa copia, o de cualquier parte de ella. Los permisos adicionales pueden ser escritos para requerir su propia eliminación bajo ciertos casos cuando se modifica la obra. Se pueden colocar permisos adicionales en material, añadidos a una obra derivada, para los cuales se establecen o se pueden establecer los permisos de derechos de autor (“copyright”) apropiados. No obstante cualquier otra disposición de esta Licencia, para el material que se añada a una obra derivada, se puede (si está autorizado por los titulares de los derechos de autor (“copyright”) del material) añadir los términos de esta Licencia con los siguientes términos:

- a. Ausencia de garantía o limitación de responsabilidad diferente de los términos de las secciones 15 y 16 de esta Licencia; o
- b. Exigir la preservación de determinados avisos legales razonables o atribuciones de autor en ese material o en los Avisos Legales Apropriados mostrados por los obras que lo contengan; o
- c. Prohibir la tergiversación del origen de ese material, o requerir que las versiones modificadas del material se marquen de maneras razonables como diferentes de la versión original; o
- d. Limitar el uso con fines publicitarios de los nombres de los licenciados o autores del material; o
- e. Negarse a ofrecer derechos concedidos por leyes de registro para el uso de alguno nombres comerciales, marcas registradas o marcas de servicio; o
- f. Exigir la compensación de los licenciados y autores de ese material por cualquiera que distribuya el material (o versiones modificadas del mismo) estableciendo obligaciones contractuales de responsabilidad sobre el destinatario, por cualquier responsabilidad que estas obligaciones contractuales impongan directamente sobre los licenciados y autores.

Todos los demás términos adicionales no permisivos son consideradas “restricciones extra” en el sentido de la sección 10. Si el Programa, tal cual se recibió, o cualquier parte del mismo, contiene un aviso indicando que se encuentra cubierto por esta Licencia junto con un término que es otra restricción, se puede quitar ese término. Si un documento de licencia contiene una restricción adicional, pero permite relicenciar o redistribuir bajo esta Licencia, se puede añadir a un material de la obra derivada bajo los términos de ese documento de licencia, a condición de que dicha restricción no sobreviva el relicenciamiento o redistribución. Si se añaden términos a una obra derivada de acuerdo con esta sección, se debe colocar, en los archivos fuente involucrados, una declaración de los términos adicionales aplicables a esos archivos, o un aviso indicando donde encontrar los términos aplicables.

Los términos adicionales, permisivos o no permisivos, pueden aparecer en forma de una licencia escrita por separado, o figurar como excepciones; los requisitos anteriores son aplicables en cualquier forma.

9. Conclusiones.

Usted no podrá propagar o modificar una obra amparada salvo lo expresamente permitido por esta Licencia. Cualquier intento diferente de propagación o modificación será considerado nulo y automáticamente se anularán sus derechos bajo esta Licencia (incluyendo las licencias de patentes concedidas bajo el tercer párrafo de la sección 11). Sin embargo, si usted deja de violar esta Licencia, entonces su licencia de un titular de los derechos de autor (“copyright”) correspondiente será

restituida (a) provisionalmente, a menos que y hasta que el titular de los derechos de autor ("copyright") explícita y finalmente termine su licencia, y (b) permanentemente, si el titular del copyright no le ha notificado su violación por algún medio razonable antes de los 60 días siguientes a la cesación.

Además, su licencia de un titular de los derechos de autor ("copyright") correspondiente será restituida permanentemente si el titular de los derechos de autor ("copyright") le notifica la violación por algún medio razonable, siendo ésta la primera vez que recibe la notificación de violación de esta Licencia (para cualquier obra) de ese titular de los derechos de autor ("copyright"), y usted subsana la violación antes de 30 días después de la recepción de la notificación.

La cancelación de sus derechos bajo esta sección no da por canceladas las licencias de terceros que hayan recibido copias o derechos de usted bajo esta Licencia. Si sus derechos han sido cancelados y no fueran renovados de manera permanente, usted no cumple los requisitos para recibir nuevas licencias para el mismo material bajo la sección 10.

10. Aceptación No Obligatoria por Tenencia de Copias.

Usted no está obligado a aceptar esta Licencia por recibir o ejecutar una copia del Programa. La propagación adicional de una obra amparada surgida únicamente como consecuencia de usar una transmisión peer-to-peer para recibir una copia tampoco requiere aceptación. Sin embargo, esta Licencia solo le otorga permiso para propagar o modificar cualquier obra amparada. Estas acciones infringen los derechos de autor ("copyright") si usted no acepta esta Licencia. Por lo tanto, al modificar o distribuir una obra amparada, usted indica que acepta esta Licencia para poder hacerlo.

11. Herencia Automática de Licencia para Destinatarios.

Cada vez que transmita una obra amparada, el destinatario recibirá automáticamente una licencia de los licenciadores originales, para ejecutar, modificar y distribuir esa obra, sujeto a esa Licencia. Usted no será responsable de asegurar el cumplimiento de esta Licencia por terceros.

Una "transacción de entidad" es una transacción que transfiere el control de una organización, o sustancialmente todos los bienes de una, o subdivide una organización, o fusiona organizaciones. Si la propagación de una obra amparada surge de una transacción de entidad, cada parte en esa transacción que reciba una copia de la obra también recibe todas las licencias de la obra que la parte interesada tuviese o pudiese ofrecer según el párrafo anterior, además del derecho a tomar posesión de las Fuentes Correspondientes de la obra a través del predecesor interesado, si el predecesor tiene o puede conseguirla con un esfuerzo razonable. Usted no podrá imponer ninguna restricción posterior en el ejercicio de los derechos otorgados o concedidos bajo esta Licencia. Por ejemplo, usted no puede imponer un pago por licencia, derechos u otros cargos por el ejercicio de los derechos otorgados bajo esta Licencia, y no puede iniciar litigios (incluyendo demandas o contrademandas en pleitos) alegando cualquier reclamación de violación de patentes por cambiar, usar, vender, ofrecer en venta o importar el Programa o alguna parte del mismo.

12. Patentes.

Un "colaborador" es un titular de los derechos de autor ("copyright") que autoriza, bajo los términos de la presente Licencia, el uso del Programa o una obra en la que se base el Programa. La obra así licenciada se denomina "versión en colaboración" del colaborador.

Las "demandas de patente esenciales" del colaborador son todas las reivindicaciones de patentes poseídas o controladas por el colaborador, ya se encuentren adquiridas o hayan sido adquiridas con posterioridad, que sean infringidas de alguna manera, permitidas por esta Licencia, al hacer, usar o vender la versión en colaboración, pero sin incluir demandas que solo sean infringidas como consecuencia de modificaciones posteriores de la versión en colaboración. Para los propósitos de esta definición, "control" incluye el derecho de conceder sublicencias de patente de forma consistente con los requisitos establecidos en la presente Licencia.

Cada colaborador le concede una licencia de la patente no-exclusiva, global y libre de regalías bajo las demandas de patente esenciales del colaborador, para hacer, usar, modificar, vender, ofrecer para venta, importar y otras formas de ejecución, modificación y difusión del contenido de la versión en colaboración.

En los siguientes tres párrafos, una "licencia de patente" se define como

cualquier acuerdo o compromiso expreso, cualquiera que sea su denominación, que no imponga una patente (como el permiso expreso para ejecutar una patente o acuerdos para no imponer demandas por infracción de patente). “Conceder” una licencias de patente de este tipo a un tercero significa hacer tal tipo de acuerdo o compromiso que no imponga una patente al tercero.

Si usted transmite una obra amparada, conociendo que está amparada por una licencia de patente, y las Fuentes Correspondientes no se encuentran disponibles de forma pública para su copia, sin cargo alguno y bajo los términos de esta Licencia, ya sea a través de un servidor público o mediante cualquier otro medio, entonces usted deberá (1) hacer que las Fuentes Correspondientes sean públicas, o (2) tratar de eliminar los beneficios de la licencia de patente para esta obra en particular, o (3) tratar de extender, de manera compatible con los requisitos de esta Licencia, la licencia de patente a terceros. “Conocer que está afectado” significa que usted tiene conocimiento real de que, para la licencia de patente, la distribución de la obra amparada en un país, o el uso de la obra amparada por sus destinatarios en un país, infringiría una o más patentes existentes en ese país que usted considera válidas por algún motivo.

Si en virtud de o en conexión con alguna transacción o acuerdo, usted transmite, o difunde con fines de distribución, una obra amparada, y concede una licencia de patente para algún tercero que reciba la obra amparada, y les autorice a usar, transmitir, modificar o difundir una copia específica de la obra amparada, entonces la licencia de patente que usted otorgue se extiende automáticamente a todos los receptores de la obra amparada y cualquier obra basada en ella.

Una licencia de patente es “discriminatoria” si no incluye dentro de su ámbito de cobertura, prohíbe el ejercicio de, o está condicionada a no ejercitar uno o más de los derechos que están específicamente otorgados por esta Licencia. Usted no debe transmitir una obra amparada si está implicado en un acuerdo con terceros que esté relacionado con el negocio de la distribución de software, en el que usted haga pagos a terceros relacionados con su actividad de distribución de la obra, bajo el que terceros conceden, a cualquier receptor de la obra amparada, una licencia de patente discriminatoria (a) en relación con las copias de la obra amparada transmitidas por usted (o copias hechas a partir de estas), o (b) principalmente para y en relación con productos específicos o compilaciones que contengan la obra amparada, a menos que usted forme parte del acuerdo, o que esa licencia de patente fuese concedida antes del 28 de marzo de 2007.

Ninguna cláusula de esta Licencia debe ser considerada como excluyente o limitante de cualquier otra licencia implicada u otras defensas legales a que pudiera tener derecho bajo la ley de propiedad intelectual vigente.

13. No Abandonar la Libertad de Otros.

Si se le imponen condiciones (bien sea por orden judicial, acuerdo o de otra manera) que contradicen las condiciones de esta Licencia, estas no le eximen de las condiciones de esta Licencia. Si usted no puede transmitir una obra amparada de forma que pueda satisfacer simultáneamente sus obligaciones bajo esta Licencia y cualesquiera otras obligaciones pertinentes, entonces, como consecuencia, usted no puede transmitirla. Por ejemplo, si usted está de acuerdo con los términos que le obligan a cobrar una regalía por la transmisión a aquellos a los que transmite el Programa, la única forma en la que usted podría satisfacer tanto esos términos como esta Licencia sería abstenerse completamente de transmitir el Programa.

14. Utilización con la Licencia Pública General Affero de GNU.

A pesar de cualquier otra disposición de esta Licencia, usted tiene permiso para enlazar o combinar cualquier obra amparada con una obra licenciada bajo la Licencia Pública General Affero de GNU en una única obra combinada, y para transmitir la obra resultante. Los términos de esta Licencia continuarán aplicándose a la parte que es la obra amparada, pero los requisitos particulares de la Licencia Pública General Affero de GNU, sección 13, concernientes a la interacción a través de una red se aplicarán a la combinación como tal.

15. Versiones Revisadas de esta Licencia.

La Free Software Foundation puede publicar versiones revisadas y/o nuevas de la Licencia General Pública de GNU de vez en cuando. Cada nueva versión será similar en espíritu a la versión actual, pero puede diferir en detalles para abordar nuevos problemas o preocupaciones.

Cada versión recibe un número de versión distintivo. Si el Programa especifica que cierta versión numerada de la Licencia General Pública de GNU “o cualquier versión posterior” se aplica a él, usted tiene la opción de seguir los términos y condiciones de esa versión numerada o de cualquier versión posterior publicada por la Free Software Foundation. Si el Programa no especifica un número de versión de la Licencia General Pública de GNU, usted puede escoger cualquier versión publicada por la Free Software Foundation.

Si el Programa especifica que un representante puede decidir que versiones futuras de la Licencia General Pública de GNU pueden ser utilizadas, la declaración pública del representante de aceptar una versión permanentemente le autoriza a usted a elegir esa versión para el Programa.

Las versiones posteriores de la licencia pueden darle permisos adicionales o diferentes. No obstante, no se impone a ningún autor o titular de los derechos de autor obligaciones adicionales como resultado de su elección de seguir una versión posterior.

16. Descargo de Responsabilidad de Garantía.

NO HAY GARANTÍA PARA EL PROGRAMA, PARA LA EXTENSIÓN PERMITIDA POR LA LEY APLICABLE. EXCEPTO CUANDO SE INDIQUE LO CONTRARIO POR ESCRITO, LOS TITULARES DE LOS DERECHOS DE AUTOR (“COPYRIGHT”) Y/O TERCEROS PROPORCIONAN EL PROGRAMA “TAL CUAL” SIN GARANTÍAS DE NINGÚN TIPO, BIEN SEAN EXPLÍCITAS O IMPLÍCITAS, INCLUYENDO, PERO NO LIMITADO A, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN Y APTITUD PARA UN PROPÓSITO PARTICULAR. EL RIESGO TOTAL EN CUANTO A CALIDAD Y RENDIMIENTO DEL PROGRAMA ES CON USTED. SI EL PROGRAMA PRESENTA ALGÚN DEFECTO, USTED ASUME EL COSTO DE TODAS LAS REVISIONES NECESARIAS, REPARACIONES O CORRECCIONES.

17. Limitación de la responsabilidad.

EN NINGÚN CASO A MENOS QUE SEA REQUERIDO POR UNA LEY APLICABLE O ACUERDO ESCRITO NINGÚN TITULAR DE LOS DERECHOS DE AUTOR (“COPYRIGHT”), O NINGÚN TERCERO QUE MODIFIQUE Y/O TRANSMITA EL PROGRAMA COMO SE PERMITE ANTERIORMENTE, SERÁ RESPONSABLE ANTE USTED POR DAÑOS, INCLUYENDO CUALESQUIERA DAÑOS GENERALES, PARTICULARES, IMPREVISTOS O DERIVADOS DEL USO O IMPOSIBILIDAD DE USO DEL PROGRAMA (INCLUYENDO, PERO NO LIMITADO A, LA PÉRDIDA DE DATOS, DATOS GENERADOS INCORRECTOS, PÉRDIDAS SUFRIDAS POR USTED O POR TERCERAS PERSONAS, O LOS FALLOS DEL PROGRAMA PARA OPERAR CON OTROS PROGRAMAS), INCLUSO SI DICHO TITULAR O UN TERCERO HA SIDO ADVERTIDO DE LA POSIBILIDAD DE TALES DAÑOS.

18. Interpretación de las Secciones 15 y 16.

Si el descargo de responsabilidad de garantía y el límite de responsabilidad proporcionado anteriormente no tiene efectos legales de acuerdo a sus términos, los juzgados deberán aplicar la ley local que más se asemeje a una renuncia absoluta de la responsabilidad civil concerniente al Programa, a menos que una garantía o una asunción de responsabilidad acompañe a la copia del Programa como resultado del pago de una tasa.

Fin de los términos y condiciones

Cómo Aplicar Estos Términos a Sus Nuevos Programas

Si desarrolla un nuevo programa, y quiere que sea lo más usado posible por el público, la mejor manera de conseguirlo es hacerlo software libre para que cualquiera pueda redistribuirlo y modificarlo bajo estos términos.

Para ello, añada la siguiente nota al programa. Lo más seguro es añadirla al principio de cada fichero fuente para declarar más efectivamente la exclusión de garantía; y cada fichero debe tener al menos la línea de “derechos de autor (“copyright”)” y un puntero a donde se pueda encontrar la anotación completa.

<una línea para dar el nombre del programa y una breve idea de lo que hace>

Copyright (C) <año> <nombre del autor>

Este programa es software libre: puede redistribuirlo y/o modificarlo bajo los términos de la Licencia General Pública de GNU publicada por la Free Software Foundation, ya sea la versión 3 de la Licencia, o (a su elección) cualquier versión posterior.

Este programa se distribuye con la esperanza de que sea útil pero SIN NINGUNA GARANTÍA; incluso sin la garantía implícita de MERCANTIBILIDAD o

604 CALIFICADA PARA UN PROPÓSITO EN PARTICULAR. Vea la Licencia General
605 Pública de GNU para más detalles.
606
607 Usted ha debido de recibir una copia de la Licencia General Pública
608 de GNU junto con este programa. Si no, vea <<http://www.gnu.org/licenses/>>.
609
610 También añada información sobre cómo contactarle por correo electrónico u
611 ordinario.
612 Si el programa es interactivo, haga que muestre un breve aviso como el
613 siguiente cuando se inicie en modo interactivo:
614 <programa> Copyright (C) <año> <nombre del autor>
615 Este programa se ofrece SIN GARANTÍA ALGUNA; escriba 'show w' para
616 consultar los detalles. Este programa es software libre, y usted puede
617 redistribuirlo bajo ciertas condiciones; escriba 'show c' para más
618 información.
619 Los hipotéticos comandos show w y show w deberán mostrar las partes
620 correspondientes de la Licencia General Pública. Por supuesto, los
621 comandos en su programa pueden ser diferentes; para una interfaz gráfica
622 de usuario, puede usar un mensaje del tipo "Acerca de".
623 También debería conseguir que su empresa (si trabaja como programador) o
624 escuela, en su caso, firme una "renuncia de derechos de autor
625 ("copyright")" sobre el programa, si fuese necesario. Para más
626 información a este respecto, y saber cómo aplicar y cumplir la licencia
627 GNU GPL, consulte <http://www.gnu.org/licenses/>.
628 La Licencia General Pública de GNU no permite incorporar sus programas
629 como parte de programas propietarios. Si su programa es una subrutina en
630 una biblioteca, podría considerar mucho más útil permitir el enlace de
631 aplicaciones propietarias con la biblioteca. Si esto es lo que quiere
632 hacer, utilice la GNU Lesser General Public License en vez de esta
633 Licencia. Pero primero, por favor consulte
634 <http://www.gnu.org/philosophy/why-not-lgpl.html>.

Apéndice D

Código del programa

D.1 Main.py

```
1  # -*- coding: utf-8 -*-
2  #!/usr/bin/env python3
3
4  '''
5      InvProy - Simulador de Redes / Proyecto de Investigación
6      https://github.com/daviddavo/InvProy
7      Copyright (C) 2016 David Davó Laviña david@ddavo.me http://ddavo.me
8
9      This program is free software: you can redistribute it and/or modify
10     it under the terms of the GNU General Public License as published by
11     the Free Software Foundation, either version 3 of the License, or
12     (at your option) any later version.
13
14     This program is distributed in the hope that it will be useful,
15     but WITHOUT ANY WARRANTY; without even the implied warranty of
16     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
17     GNU General Public License for more details.
18
19     You should have received a copy of the GNU General Public License
20     along with this program. If not, see <http://www.gnu.org/licenses/>.
21
22     //////////////////////////////////
23
24     Este programa es código libre: Puedes redistribuirlo y/o modificarlo
25     bajo los términos de la licencia GNU General Public License tal y como
26     publicado por la Free Software Foundation, ya sea la versión 3 de layout
27     licencia o la más reciente.
28
29     Este programa es distribuido con la esperanza de que sea útil, pero
30     SIN NINGUNA GARANTÍA; sin siquiera la garantía implícita de COMERCIABILIDAD
31     o de la APTITUD DE LA MISMA PARA UN PROPÓSITO PARTICULAR. Ver la GNU General
32     Public License para más detalles.
33
34     Debes haber recibido una copia de la GNU General Public License con
35     este programa, si no es así, ver <http://www.gnu.org/licenses/>.
36 '''
37 from datetime import datetime
38 startTime = datetime.now()
39
40 import configparser, os, csv, sys, time, random, math
41 import xml.etree.ElementTree as xmltree
42 from ipaddress import ip_address
43 from random import choice
44
45 #Esto hace que el programa se pueda ejecutar fuera de la carpeta.
46 startcwd = os.getcwd()
47
48 try:
49     os.chdir(os.path.dirname(sys.argv[0]))
50 except:
51     pass
52
53 os.system("clear")
54 print("\033[91m#####\033[00m")
```

```

55
56 print("InvProxy Copyright (C) 2016 David Davó Laviña\ndavid@ddavo.me <http://ddavo.me>\n\
57 This program comes with ABSOLUTELY NO WARRANTY; for details go to 'Ayuda > Acerca de'\n\
58 This is free software, and you are welcome to redistribute it\n\
59 under certain conditions\n")
60
61 try: #Intenta importar los modulos necesarios
62     #sys.path.append("Modules/")
63     import Modules.Test
64 except:
65     print("Error: No se han podido importar los modulos...")
66     sys.exit()
67
68 #Aqui importamos los modulos del programa que necesitamos...
69
70 from Modules.logmod import *
71 from Modules import save
72
73 def lprint(*objects, sep=" ", end="\n", file=sys.stdout, flush=False):
74     print(*objects, sep=sep, end=end, file=file, flush=flush)
75     thing=str()
76     for i in objects:
77         thing += str(i) + sep
78     writeonlog(thing)
79
80 lprint("Start loading time: " + time.strftime("%H:%M:%S"))
81
82 try:
83     #Importando las dependencias de la interfaz
84     import gi
85     gi.require_version('Gtk', '3.0')
86     from gi.repository import Gtk, GObject, Gdk, GdkPixbuf
87 except:
88     lprint("Por favor, instala PyGObject en tu ordenador. \n En ubuntu suele ser 'apt-get install python3-gi'\n
89         ↪ En Archlinux es 'pacman -S python-gobject'")
90     sys.exit()
91
92 try:
93     import cairo
94 except:
95     print("Necesitas tener instalado cairo")
96     print("Como es lógico, pon 'pacman -S python-cairo' en Archlinux")
97     sys.exit()
98
99 #Definiendo un par de cosillas necesarias
100
101 gtk = Gtk
102 config = configparser.RawConfigParser()
103 configdir = "Config.ini"
104 config.read(configdir)
105 allobjects = []
106
107 #Funcion que convierte un numero a una str con [digits] cifras
108 def digitsnumber(number, digits):
109     if len(str(number)) == digits:
110         return str(number)
111     elif len(str(number)) < digits:
112         return "0" * (digits - len(str(number))) + str(number)
113     else:
114         return "-1"
115
116 #Convierte hexadecimal a RGBA tal y como Gdk lo requiere
117 def hex_to_rgba(value):
118     value = value.lstrip('#')
119     if len(value) == 3:
120         value = ''.join([v*2 for v in list(value)])
121     (r1,g1,b1,a1)=tuple(int(value[i:i+2], 16) for i in range(0, 6, 2))+1,)
122     (r1,g1,b1,a1)=(r1/255.00000,g1/255.00000,b1/255.00000,a1)
123
124     return (r1,g1,b1,a1)

```

```

124
125 print("#42FF37", hex_to_rgba("#42FF37"))
126
127 #Comprueba la integridad del pack de recursos
128 def checkres(recurdir):
129     files = ["Cable.png", "Router.png", "Switch.png", "Computer.png", "Hub.png"]
130     cnt = 0
131     ss = []
132     for i in files:
133         if os.path.isfile(recurdir + i):
134             cnt += 1
135         else:
136             ss.append(i)
137
138     if not (cnt == len(files)):
139         lprint("WARNING!!!!111!!!!111!")
140         lprint("Faltan archivos en resources/"+recurdir)
141         lprint(ss)
142         sys.exit()
143     else:
144         lprint("Estan todos los archivos")
145
146 checkres(config.get("DIRS", "respack"))
147
148 #Envia a la Statusbar informacion.
149 contador = 0
150 def push_elemento(texto):
151     global contador
152     varra1 = builder.get_object("barra1")
153     data = varra1.get_context_id("Ejemplocontextid")
154     testo = time.strftime("%H:%M:%S") + " | " + texto
155     contador = contador + 1
156     varra1.push(data, testo)
157     writeonlog(texto)
158
159 #Retorna un entero en formato de bin fixed
160 def bformat(num, fix):
161     if type(num) == int:
162         return str(("{0:0}" + str(fix) + "b").format(num))
163     else:
164         return "ERROR"
165
166 gladefile = "Interface2.glade"
167
168 try:
169     builder = Gtk.Builder()
170     builder.add_from_file(gladefile)
171     writeonlog("Cargando interfaz")
172     lprint("Interfaz cargada\nCargados un total de " + str(len(builder.get_objects())) + " objetos")
173     xmlroot = xmltree.parse(gladefile).getroot()
174     lprint("Necesario Gtk+ " + xmlroot[0].attrib["version"]+".0", end="")
175     lprint(" | Usando Gtk+ "
176           ↪ " +str(Gtk.get_major_version())+"."+str(Gtk.get_minor_version())+"."+str(Gtk.get_micro_version()))
177 except Exception as e:
178     lprint("Error: No se ha podido cargar la interfaz.")
179     if "required" in str(e):
180         xmlroot = xmltree.parse(gladefile).getroot()
181         lprint("Necesario Gtk+ " + xmlroot[0].attrib["version"]+".0", end="\n")
182         lprint(">Estas usando
183           ↪ Gtk+ "+str(Gtk.get_major_version())+"."+str(Gtk.get_minor_version())+"."+str(Gtk.get_micro_version()))
184     else:
185         lprint("Debug:", e)
186         sys.exit()
187
188 #Intenta crear el archivo del log
189 createlogfile()
190
191 #CONFIGS
192 WRES, HRES = int(config.get("GRAPHICS", "WRES")), int(config.get("GRAPHICS", "HRES"))

```

```

192 resdir      = config.get("DIRS", "respack")
193
194 lprint(resdir)
195
196 #CLASSES
197
198 allkeys = set()
199 cables = []
200 clickedobjects = set() #Creamos una cosa para meter los ultimos 10 objetos clickados. (EN DESUSO)
201 clicked = 0
202 bttnclicked = 0
203 areweputtingcable = 0
204
205 #Función a medias, esto añadirá un objeto a la cola de ultimos objetos clickados, por si luego queremos deshacerlo
    ↪ o algo.
206 def appendtoclicked(objeto):
207     clickedobjects.insert(0, objeto)
208     try:
209         clickedobjects.remove(9)
210     except:
211         pass
212
213 class MainClase(Gtk.Window):
214     def __init__(self):
215         global resdir
216
217         self.ventana = builder.get_object("window1")
218         self.ventana.connect("key-press-event", self.on_key_press_event)
219         self.ventana.connect("key-release-event", self.on_key_release_event)
220         self.ventana.set_default_size(WRES, HRES)
221         self.ventana.set_keep_above(bool(config.getboolean("GRAPHICS", "window-set-keep-above")))
222
223         builder.get_object("Revealer1").set_reveal_child(bool(config.getboolean("GRAPHICS",
    ↪ "revealer-show-default")))
224
225         i = int(config.get('GRAPHICS', 'toolbutton-size'))
226
227         #Probablemente estas dos variables se puedan coger del builder de alguna manera, pero no se cómo.
228         start = 3
229         end = 8
230         jlist = ["Router.png", "Switch.png", "Cable.png", "Computer.png", "Hub.png"]
231         for j in range(start, end):
232             objtmp = builder.get_object("toolbutton" + str(j))
233             objtmp.connect("clicked", self.toolbutton_clicked)
234             objtmp.set_icon_widget(Gtk.Image.new_from_pixbuf(Gtk.Image.new_from_file(resdir +
    ↪ jlist[j-start]).get_pixbuf().scale_simple(i, i, GdkPixbuf.InterpType.BILINEAR)))
235             objtmp.set_tooltip_text(jlist[j - start].replace(".png", ""))
236
237         global configWindow
238         #configWindow = cfgWindow()
239
240         builder.get_object("imagemenuitem1").connect("activate", self.new)
241         builder.get_object("imagemenuitem9").connect("activate", self.showcfgwindow)
242         builder.get_object("imagemenuitem1").connect("activate", self.new)
243         builder.get_object("imagemenuitem3").connect("activate", self.save)
244         builder.get_object("imagemenuitem4").connect("activate", self.save)
245         builder.get_object("imagemenuitem2").connect("activate", self.load)
246         builder.get_object("imagemenuitem10").connect("activate", about().show)
247         builder.get_object("show_grid").connect("toggled", self.togglegrid)
248
249         ### EVENT HANDLERS###
250
251         handlers = {
252             "onDeleteWindow":      exiting,
253             "onExitPress":         exiting,
254             "onRestartPress":      restart,
255
256         }
257         builder.connect_signals(handlers)
258

```

```

259         builder.get_object("toolbutton1").connect("clicked", objlst.show)
260
261         self.ventana.show_all()
262
263     class ObjLst():
264         def __init__(self):
265             self.view = builder.get_object("objetos_treeview")
266             self.tree = Gtk.TreeStore(str,str)
267             renderer = Gtk.CellRendererText()
268             column = Gtk.TreeViewColumn("Objetos", renderer, text=0)
269             self.view.append_column(column)
270             column.set_sort_column_id(0)
271
272             renderer = Gtk.CellRendererText()
273             column = Gtk.TreeViewColumn("Valor", renderer, text=1)
274             column.set_sort_column_id(1)
275             self.view.append_column(column)
276             self.view.set_model(self.tree)
277             self.view.show_all()
278
279             self.revealer = builder.get_object("Revealer1")
280             print("Revealer:",self.revealer.get_reveal_child())
281             self.panpos = 100
282
283         def append(self, obj, otherdata=[]):
284             #SI OBJ YA ESTÁ, QUE AÑADA ATRIBUTOS A LA LISTA.
285             it1 = self.tree.append(None, row=[obj.name, obj.objecttype])
286             it2 = self.tree.append(it1, row=["MAC", str(obj.madir)])
287             itc = self.tree.append(it1, row=["Conexiones", "{}/{ {}".format(len(obj.connections),
288                                     ↪ obj.max_connections)])
289             for i in otherdata:
290                 self.tree.append(it1, row=i)
291
292             obj.trdic = {"MAC":it2, "Connections":itc}
293
294             return it1
295
296         def update(self, obj, thing, val):
297             if thing in obj.trdic.keys():
298                 self.tree.set_value(obj.trdic[thing], 1, val)
299             else:
300                 it = self.tree.append(obj.trlst, row=[thing, val])
301                 obj.trdic[thing] = it
302
303         def upcon(self, obj):
304             if not hasattr(obj, "trcondic"):
305                 obj.trcondic = {}
306             #objlst.tree.append(self.trdic["Connections"], row=[self.name, self.objecttype])
307             self.tree.set_value(obj.trdic["Connections"], 1, "{}/{ {}".format(len(obj.connections),
308                                     ↪ obj.max_connections))
309             for i in obj.connections:
310                 print(i.__repr__(), obj.trcondic)
311                 if i in obj.trcondic.keys():
312                     self.tree.set_value(obj.trcondic[i], 0, i.name)
313                 else:
314                     r = self.tree.append(obj.trdic["Connections"], row=[i.name, ""])
315                     obj.trcondic[i] = r
316
317         def show(self, *args):
318             rev = self.revealer.get_reveal_child()
319             if rev:
320                 self.panpos = builder.get_object("paned1").get_position()
321
322             builder.get_object("paned1").set_position(-1)
323             self.revealer.set_reveal_child(not self.revealer.get_reveal_child())
324
325             if not rev:
326                 pass
327
328         def set_value(self,*args):

```

```

327         self.tree.set_value(*args)
328
329     def delete(self, obj):
330         self.tree.remove(obj.trlst)
331
332     def showcfgwindow(self, *args):
333         global configWindow
334         try:
335             configWindow.show()
336         except:
337             configWindow = cfgWindow()
338             configWindow.show()
339
340     #24/06 Eliminada startCable(), incluida en toolbutton_clicked
341
342     def togglegrid(self, *widget):
343         widget = widget[0]
344         global TheGrid
345         obj = TheGrid.backgr_lay
346         if widget.get_active() != True and obj.is_visible():
347             obj.hide()
348         else:
349             obj.show()
350
351     #Una función para gobernarlos a todos.
352     def toolbutton_clicked(self, objeto):
353         global clicked
354         global bttnclicked
355         global areweputtingcable
356         if areweputtingcable != 0:
357             areweputtingcable = 0
358             push_elemento("Cancelada acción de poner un cable")
359
360         if objeto.props.label == "toolbutton5":
361             lprint("Y ahora deberíamos poner un cable")
362             push_elemento("Ahora pulsa en dos objetos")
363             areweputtingcable = "True"
364
365         object_name = objeto.props.label
366         clicked = True
367         bttnclicked = object_name
368
369     #Al pulsar una tecla registrada por la ventana, hace todo esto.
370     def on_key_press_event(self, widget, event):
371         keyname = Gdk.keyval_name(event.keyval).upper() #El upper es por si está BLOQ MAYUS activado.
372         global allkeys #Esta es una lista que almacena todas las teclas que están siendo pulsadas
373         if config.getboolean("BOOLEANS", "print-key-pressed") == True:
374             lprint("Key %s (%d) pulsada" % (keyname, event.keyval))
375             lprint("Todas las teclas: ", allkeys)
376         if not keyname in allkeys:
377             allkeys.add(keyname)
378         if ("CONTROL_L" in allkeys) and ("Q" in allkeys):
379             exiting(1)
380         if ("CONTROL_L" in allkeys) and ("R" in allkeys):
381             restart()
382         if ("CONTROL_L" in allkeys) and ("U" in allkeys):
383             global allobjects
384             print("HARD UPDATE")
385             print(allobjects)
386             for obj in allobjects:
387                 obj.update()
388
389         if ("CONTROL_L" in allkeys) and ("S" in allkeys):
390             global allobjects
391             MainClase.save()
392         if ("CONTROL_L" in allkeys) and ("L" in allkeys):
393             MainClase.load()
394             allkeys.discard("CONTROL_L")
395             allkeys.discard("L")
396

```

```

397         #Para no tener que hacer click continuamente
398         if ("Q" in allkeys):
399             self.toolbutton_clicked(builder.get_object("toolbutton3"))
400         if "W" in allkeys:
401             self.toolbutton_clicked(builder.get_object("toolbutton4"))
402         if "E" in allkeys:
403             self.toolbutton_clicked(builder.get_object("toolbutton5"))
404         if "R" in allkeys:
405             self.toolbutton_clicked(builder.get_object("toolbutton6"))
406         if "T" in allkeys:
407             self.toolbutton_clicked(builder.get_object("toolbutton7"))
408         return keyname
409
410     #Al dejar de pulsar la tecla deshace lo anterior.
411     def on_key_release_event(self, widget, event):
412         keynameb = Gdk.keyval_name(event.keyval).upper()
413         if config.getboolean("BOOLEANS", "print-key-pressed") == True:
414             lprint("Key %s (%d) released" % (keynameb, event.keyval))
415         global allkeys
416         allkeys.discard(keynameb)
417
418     def drag_drop(widget, context, x, y, time):
419         push_elemento("Drag drop at " + str(x) + "," + str(y) )
420
421     #Comprueba si el objeto tiene una ip asignada
422     def has_ip(self):
423         try:
424             if self.IP != None:
425                 return True
426             else:
427                 return False
428         except:
429             return False
430
431     def save(*args):
432         global cables
433         global allobjects
434         lscl = 0
435         try:
436             if args[1].get_label() == "gtk-save-as":
437                 print("Guardando como")
438                 lscl = 1
439         except:
440             pass
441         save.save(allobjects,cables, aslc=lscl)
442         push_elemento("Guardando...")
443     def load(*args):
444         global cables
445         global allobjects
446         save.load(allobjects,cables)
447         push_elemento("Cargando...")
448     def new(*args):
449         global allobjects
450         global cables
451         save.last = 0
452         while len(allobjects) > 0:
453             allobjects[0].delete(pr=0)
454         while len(cables) > 0:
455             cables[0].delete()
456
457     def new(*args):
458         global cables
459         global allobjects
460         while len(allobjects) > 0:
461             allobjects[0].delete(pr=0)
462
463     #Esta clase no es mas que un prompt que pide 'Si' o 'No'.
464     #La función run() retorna 1 cuando se clicka sí y 0 cuando se clicka no, así sirven como enteros y booleans.
465     class YesOrNoWindow(Gtk.Dialog):
466         def __init__(self, text, *args, Yest="Sí", Not="No"):

```



```

467
468     self.builder = Gtk.Builder()
469     self.builder.add_from_file(gladefile)
470
471     self.yesornowindow = self.builder.get_object("YesOrNoWindow")
472     self.labeldialog = self.builder.get_object("YoN_label")
473     self.nobutton = self.builder.get_object("YoN_No")
474     self.yesbutton = self.builder.get_object("YoN_Yes")
475
476     self.nobutton.connect("clicked", self.on_button_clicked)
477     self.yesbutton.connect("clicked", self.on_button_clicked)
478
479     self.labeldialog.set_text(text)
480     self.yesbutton.set_label(Yes)
481     self.nobutton.set_label(Not)
482
483     self = self.yesornowindow
484
485     def on_button_clicked(self, widget):
486         dialog = self
487
488     def run(self):
489         return self.yesornowindow.run()
490         self.yesornowindow.hide()
491
492     def destroy(self):
493         self.yesornowindow.destroy()
494
495 objetocable1 = None
496
497 #Esto es el Grid donde van las cosicas. A partir de aqui es donde esta lo divertido.
498 class Grid():
499     def __init__(self):
500         #16/06/16 MAINPORT PASA A SER VARIAS LAYERS
501         self.overlay = builder.get_object("overlay1")
502         self.mainport = Gtk.Layout.new()
503         self.cables_layer = Gtk.Layout.new()
504         self.backgr_layer = Gtk.Layout.new()
505         self.select_layer = Gtk.Layout.new() #Aparecer un fondo naranja en la cuadrícula cuando se selcciona un objeto
506         self.animat_layer = Gtk.Layout.new() #La capa de las animaciones de los cables
507         self.overlay.add_overlay(self.backgr_layer)
508         self.overlay.add_overlay(self.select_layer)
509         self.overlay.add_overlay(self.cables_layer)
510         self.overlay.add_overlay(self.animat_layer)
511         self.overlay.add_overlay(self.mainport)
512
513         self.viewport = builder.get_object("viewport1")
514         self.eventbox = builder.get_object("eventbox1")
515         self.eventbox.connect("button-press-event", self.clicked_on_grid)
516         #self.viewport.get_hadjustment().set_value(800)
517
518         self.wres = config.getint("GRAPHICS", "viewport-wres")
519         self.hres = config.getint("GRAPHICS", "viewport-hres")
520         self.sqres = config.getint("GRAPHICS", "viewport-sqres")
521         self.overlay.set_size_request(self.wres*self.sqres, self.hres*self.sqres)
522
523         #Modifica el color de fondo del viewport
524         clr = hex_to_rgba(config.get("GRAPHICS", "viewport-background-color"))
525         print("CLR:", clr)
526         self.viewport.override_background_color(Gtk.StateFlags.NORMAL, Gdk.RGBA(*clr))
527
528         #13/07/16 Ahora esto va por cairo, mejooor.
529         ### INICIO CAIRO
530
531         width, height, sq = self.wres*self.sqres, self.hres*self.sqres, self.sqres
532         surface = cairo.ImageSurface(cairo.FORMAT_ARGB32, width, height)
533         ctx = cairo.Context(surface)
534         ctx.close_path ()
535         ctx.set_source_rgba(0,0,0,1)
536         ctx.set_line_width(1)

```

```

537
538     for i in range(self.wres):
539         ctx.move_to(i*sq, 0)
540         ctx.line_to(i*sq, height)
541     for i in range(self.hres):
542         ctx.move_to(0, i*sq)
543         ctx.line_to(width, i*sq)
544
545
546     ctx.stroke()
547     self.image = Gtk.Image.new_from_surface(surface)
548     ### FINAL DE LO DE CAIRO
549
550     self.backgr_lay.put(self.image, 0, 0)
551
552     def subshow(widget):
553         #Para que no aparezca arriba a la izquierda:
554         scrolled = builder.get_object("scrolledwindow1")
555         scrolled.get_vadjustment().set_value(height/3)
556         scrolled.get_hadjustment().set_value(width/3)
557
558     if config.getboolean("GRAPHICS","start-centered"):
559         builder.get_object("window1").connect("show", subshow)
560     self.overlay.show_all()
561     self.contadorback = 0
562
563     def moveto(self, image, x, y, *args, layout=None):
564         if x < self.wres and y < self.hres:
565             if layout == None:
566                 layout = self.mainport
567             elif str(layout.__class__.__name__) == "Layout":
568                 layout = layout
569             else:
570                 print("layout.__class__.__name__", layout.__class__.__name__)
571             if image in layout.get_children():
572                 layout.move(image, x*self.sqres, y*self.sqres)
573             else:
574                 layout.put(image, x*self.sqres, y*self.sqres)
575         else:
576             print("\033[31mError: Las coordenadas se salen del grid\033[00m")
577
578     def clicked_on_grid(self, widget, event, *args):
579         global clicked
580         global bttnclicked
581         global allobjects
582         global areweputtingcable
583         self.contadorback += 1
584
585         push_elemento("Clicked on grid @" + str(self.gridparser(event.x, self.wres)) + "," +
586             ↪ str(self.gridparser(event.y, self.hres)))
587
588         if self.searchforobject(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres)) == False:
589             if clicked == 1:
590                 push_elemento("Clicked: " + str(clicked) + " bttnclicked: " + str(bttnclicked))
591                 if bttnclicked == "Router":
592                     Router(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres))
593                     push_elemento("Creado objeto router")
594                 elif bttnclicked == "toolbutton4":
595                     Switch(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres))
596                     push_elemento("Creado objeto switch")
597                 elif bttnclicked == "toolbutton6":
598                     Computador(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres))
599                     push_elemento("Creado objeto Computador")
600                 elif bttnclicked == "toolbutton7":
601                     Hub(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres))
602                     push_elemento("Creado objeto Hub")
603             elif self.searchforobject(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres)) !=
604                 ↪ False:
605                 push_elemento("Ahí ya hay un objeto, por favor selecciona otro sitio")

```

```

605     else:
606         lprint("pls rebisa l codigo")
607     clicked = 0
608     bttnclicked = 0
609
610     #Button: 1== Lclick, 2== Mclick
611     #Para comprobar si es doble o triple click: if event.type == gtk.gdk.BUTTON_PRESS, o gtk.gdk_2_BUTTON_PRESS
612     if event.button == 3:
613         rclick_Object = self.searchforobject(self.gridparser(event.x, self.wres), self.gridparser(event.y,
614             ↪ self.hres))
615         if rclick_Object != False:
616             rclick_Object.rclick(event)
617         else:
618             print("Agua")
619
620     if areweputtingcable != 0:
621         objeto = self.searchforobject(self.gridparser(event.x, self.wres), self.gridparser(event.y, self.hres))
622         if objeto == False:
623             push_elemento("Selecciona un objeto por favor")
624         elif objeto != False:
625             if len(objeto.connections) < objeto.max_connections:
626                 if areweputtingcable == "True":
627                     push_elemento("Ahora selecciona otro más")
628                     areweputtingcable = "Secondstep"
629                     global objetocable1
630                     objetocable1 = objeto
631                 elif areweputtingcable == "Secondstep":
632                     push_elemento("Poniendo cable")
633                     areweputtingcable = 0
634                     global objetocable1
635                     cable = Cable(objetocable1, objeto)
636                     objeto.connect(objetocable1, cable)
637                     objetocable1 = 0
638
639             else:
640                 push_elemento("Número máximo de conexiones alcanzado")
641
642     #Te pasa las cordenadas int que retorna Gtk a coordenadas del Grid, bastante sencillito. Tienes que llamarlo 2
643     ↪ veces, una por coordenada
644     def gridparser(self, coord, cuadrados, mode=0):
645         if mode == 0:
646             partcoord = coord / self.sqres
647             for i in range(cuadrados + 1):
648                 if partcoord < i:
649                     return i
650             else:
651                 pass
652         if mode == 1:
653             return coord * self.sqres
654
655     def resizetogrid(self, image):
656         #Image debe ser una imagen gtk del tipo gtk.Image
657         pixbuf = image.get_pixbuf()
658         pixbuf = pixbuf.scale_simple(self.sqres, self.sqres, GdkPixbuf.InterpType.BILINEAR)
659         image.set_from_pixbuf(pixbuf)
660
661     #Una función para encontrarlos,
662     def searchforobject(self, x, y):
663         global allobjects
664         localvar = False
665         for i in range(len(allobjects)):
666             if allobjects[i].x == x:
667                 if allobjects[i].y == y:
668                     localvar = True
669                     objeto = allobjects[i]
670                     break
671         if localvar == True:
672             return objeto
673         else:
674             return False

```

```

673
674     def __str__(self):
675         lprint("No se que es esto")
676
677 TheGrid = Grid()
678
679 #Clases de los distintos objetos. Para no escribir demasiado tenemos la clase ObjetoBase
680 #De la que heredaran las demas funciones
681 cnt_objects = 1
682 cnt_rows = 2
683 objlst = MainClase.ObjLst()
684
685 import uuid
686
687 class ObjetoBase():
688     allobjects = []
689     cnt = 0
690     #Una función para atraerlos a todos y atarlos en las tinieblas
691     def __init__(self, x, y, objtype, *args, name="Default", maxconnections=4, ip=None):
692         global cnt_objects
693         global cnt_rows
694         global allobjects
695         global gladefile
696
697         #IMPORTANTE: GENERAR UUID PARA CADA OBJETO
698         #La v4 crea un UUID de forma aleatoria
699         self.uuid = uuid.uuid4()
700         print("\033[96mUUID:\033[00m", self.uuid)
701
702         self.builder = Gtk.Builder()
703         self.builder.add_from_file(gladefile)
704         self.menueemergente = self.builder.get_object("grid_rclick")
705         self.builder.get_object("grid_rclick-disconnect_all").connect("activate", self.disconnect)
706         self.builder.get_object("grid_rclick-delete").connect("activate", self.delete)
707         self.builder.get_object("grid_rclick-debug").connect("activate", self.debug)
708
709         allobjects.append(self)
710
711         self.realx = x * TheGrid.sqres
712         self.realy = y * TheGrid.sqres
713         self.x = x -1
714         self.y = y -1
715         self.connections = []
716         self.cables = []
717         self.max_connections = maxconnections
718
719         #Algún día pasaré todos los algoritmos a algoritmos de búsqueda binaria
720         for f in os.listdir(resdir):
721             lprint(f, f.startswith(objtype))
722             if f.startswith(objtype) and ( f.endswith(".jpg") or f.endswith(".png") ):
723                 self.imgdir = resdir + f
724                 break
725
726         self.image = gtk.Image.new_from_file(self.imgdir)
727         self.resizetogrid(self.image)
728         if name == "Default" or name == None:
729             self.name = self.objtype + " " + str(self.__class__.cnt)
730         else:
731             self.name = name
732         cnt_objects += 1
733         self.__class__.cnt += 1
734
735         TheGrid.moveto(self.image, self.x, self.y)
736         self.image.show()
737
738         self.macdir = mac()
739
740         print("MAC:", self.macdir, int(self.macdir), bin(self.macdir))
741         if ip == None:
742             print("No ip definida")

```

```

743         self.ipstr = "None"
744
745         #Ahora vamos con lo de aparecer en la lista de la izquierda,
746         #aunque en realidad es un grid
747         lista = objlst
748         self.trlst = lista.append(self)
749         self.image.set_tooltip_text(self.name + " (" + str(len(self.connections)) + "/" +
750             ↪ str(self.max_connections) + ")\n" + self.ipstr)
751
752         self.window_changethings = w_changethings(self)
753         self.builder.get_object("grid_rclick-name").connect("activate", self.window_changethings.show)
754
755         self.cnt = 0 #Se me olvido que hace esta cosa
756
757     def load(self):
758         global cnt_objects
759         global cnt_rows
760         global allobjects
761         self.builder = Gtk.Builder()
762         self.builder.add_from_file(gladefile)
763         self.menueemergente = self.builder.get_object("grid_rclick")
764         self.builder.get_object("grid_rclick-disconnect_all").connect("activate", self.disconnect)
765         self.builder.get_object("grid_rclick-delete").connect("activate", self.delete)
766         self.builder.get_object("grid_rclick-debug").connect("activate", self.debug)
767         self.connections = []
768         self.cables = []
769         cnt_objects += 1
770         self.__class__.cnt += 1
771         allobjects.append(self)
772         self.image = gtk.Image.new_from_file(self.imgdir)
773         self.resizetogrid(self.image)
774         TheGrid.moveto(self.image, self.x-1, self.y-1)
775         self.image.show()
776
777         self.trlst = objlst.append(self)
778
779         self.image.set_tooltip_text(self.name + " (" + str(len(self.connections)) + "/" +
780             ↪ str(self.max_connections) + ")\n" + self.ipstr)
781         self.window_changethings = w_changethings(self)
782         self.builder.get_object("grid_rclick-name").connect("activate", self.window_changethings.show)
783
784         print("CABLES",self.cables)
785
786     #Esta funcion retorna una str cuando se usa el objeto. En lugar de <0XXXXXXXX object>
787     def __str__(self):
788         return "<Tipo: " + self.objecttype + " | Name: " + self.name + " | Pos: " + str(self.x) + ", " + str(self.y)
789             ↪ + ">"
790
791     def debug(self, *args):
792         print("DEBUG")
793         print("MAC:", self.macdir, int(self.macdir))
794
795     def rclick(self, event):
796         global rclick_Object
797         rclick_Object = self
798
799         print(self)
800         lprint("rclick en", self.x, self.y, self.objecttype, "\nConnections: ", end="")
801         lprint(self.connections)
802         self.rmenu = self.menueemergente
803         if self.objecttype == "Computer" and len(self.compcon()) > 0:
804             self.builder.get_object("grid_rclick-sendpkg").show()
805         else:
806             self.builder.get_object("grid_rclick-sendpkg").hide()
807         if len(self.connections) > 0:
808             self.builder.get_object("grid_rclick-disconnect").show_all()
809         else:
810             self.builder.get_object("grid_rclick-disconnect").hide()
811         self.rmenu.popup(None, None, None, None, event.button, event.time)

```

```

810 def resizetogrid(self, image, *args):
811     #Ver resizetogrid en Grid (clase)
812     lprint(*args)
813     TheGrid.resizetogrid(image)
814
815 def clickado(self, widget, event):
816     lprint("Clickado en objeto " + str(self) + " @ " + str(self.x) + ", " + str(self.y))
817
818     #Esta función se encarga de comprobar a que ordenador(es) está conectado
819     #en total, pasando por routers, hubs y switches.
820
821     #Nota, hacer que compruebe que ordenadores tienen IP, y cuales no.
822 def compcon(self, *args):
823     passedyet = []
824     comps = []
825     reself = self
826
827     def subcompcon(notself, *args):
828         nonlocal passedyet
829         nonlocal reself
830         subcomps = []
831
832         interc = notself.connections
833         #print(notself, "connections:", interc)
834         #next(interc)
835
836         for con in interc:
837             if con.uuid != reself.uuid and con.uuid not in [obj.uuid for obj in passedyet]:
838                 passedyet.append(con)
839                 #print(con)
840                 if con.objecttype == "Computer":
841                     subcomps.append(con)
842                 elif con.objecttype == "Switch" or con.objecttype == "Hub":
843                     subcomps.extend(subcompcon(con))
844                 else:
845                     print("Saltado", con)
846                     pass
847                 #passedyet.append(con)
848
849             #print("passedyet", passedyet)
850             return subcomps
851
852     comps.extend(subcompcon(self))
853
854     try:
855         #comps.remove(self)
856         pass
857     except:
858         pass
859
860     if args == 1 or "Gtk" in str(args):
861         print("Comps:", comps)
862         print("\nCompsname:", [x.name for x in comps])
863
864     return comps
865
866     #Comprueba si un objeto está conectado a otro.
867 def isconnected(self, objeto):
868     cons = compcon(self)
869     if objeto in cons:
870         return True
871     else:
872         return False
873
874     #TODO: Para no tener que actualizar todo, que compruebe el que cambió
875     #TODO: !! Hacer que modifique el menu_emergente (Hecho a medias xds)
876     #Nota !!: No puedes buscar un objeto en una lista, debes buscar sus atr.
877 def update(self):
878     print("\033[95m>>Updating\033[00m", self)
879     print(self.builder.get_object("grid_rclick-disconnect"))

```

```

880     self.image.set_tooltip_text(self.name + " (" + str(len(self.connections)) + "/" +
    ↪     str(self.max_connections) + ")")
881     objlst.set_value(self.trlst, 0, self.name)
882
883     objlst.update(self,"MAC", str(self.macdir))
884     for child in self.builder.get_object("grid_rclick-disconnect").get_submenu().get_children():
885         if child.props.label.upper() != "TODOs":
886             if child.link.uuid not in [x.uuid for x in self.connections]:
887                 print("Object", child.link.__repr__(), "in connections", self.connections)
888                 child.hide()
889                 child.destroy()
890             else:
891                 print("Object", child.link.__repr__(), "in self.connections", self.connections)
892         pass
893
894     objlst.upcon(self)
895
896     print("\033[95m<<\033[00m")
897
898     def connect(self, objeto, cable):
899         tmp = Gtk.MenuItem.new_with_label(objeto.name)
900         self.builder.get_object("grid_rclick-disconnect").get_submenu().append(tmp)
901         tmp.show()
902         tmp.connect("activate", self.disconnect)
903         #link es un objeto vinculado al widget, luego es útil.
904         tmp.link = objeto
905         tmp2 = Gtk.MenuItem.new_with_label(objeto.name)
906
907         if self.__class__.__name__ != "Switch" and self.__class__.__name__ != "Hub":
908             tmp2.connect("activate", self.send_pck)
909             tmp2.show()
910             tmp2.link = objeto
911
912         tmp = Gtk.MenuItem.new_with_label(self.name)
913         objeto.builder.get_object("grid_rclick-disconnect").get_submenu().append(tmp)
914         tmp.show()
915         tmp.connect("activate", objeto.disconnect)
916         tmp.link = self
917         tmp2 = Gtk.MenuItem.new_with_label(self.name)
918
919         if objeto.__class__.__name__ != "Switch" and objeto.__class__.__name__ != "Hub":
920             tmp2.show()
921             tmp2.connect("activate", objeto.send_pck)
922             tmp2.link = self
923
924         self.connections.append(objeto)
925         self.cables.append(cable)
926         #objlst.tree.append(self.trdic["Connections"], row=[objeto.name, objeto.objectype])
927
928         objeto.connections.append(self)
929         objeto.cables.append(cable)
930         #objlst.tree.append(objeto.trdic["Connections"], row=[self.name, self.objectype])
931
932         self.update()
933         objeto.update()
934
935         if objeto.__class__.__name__ == "Switch":
936             print("Connecting {} to {}".format(objeto, self))
937             objeto.connectport(self)
938         if self.__class__.__name__ == "Switch":
939             print("Connecting {} to {}".format(objeto, self))
940             self.connectport(objeto)
941
942     def disconnect(self, widget, *args, de=None):
943         print("Cables:", self.cables)
944         #QUICKFIX
945         try:
946             if widget.props.label.upper() == "TODOs" and de == None:
947                 de = "All"
948             elif de == None:

```

```

949         de = widget.link
950     except:
951         print("NO WIDGET AT DISCONNECT()")
952
953     if de == "All":
954         ###NO FUNCIONA DEL TODO BIEN, NO USAR###
955         #Bug, el ultimo cable no se borra
956         print("Ahora a desconectar de todos")
957         while len(self.connections) > 0:
958             self.disconnect(widget, de=self.connections[0])
959
960     else:
961         objlst.tree.remove(self.trcondic[de])
962         del self.trcondic[de]
963         objlst.tree.remove(de.trcondic[self])
964         del de.trcondic[self]
965
966         de.connections.remove(self)
967         self.connections.remove(de)
968
969         iterc = iter(self.builder.get_object("grid_rclick-disconnect").get_submenu().get_children())
970         next(iterc)
971         print("\033[91mLinks\033[00m", [x.link for x in iterc])
972
973         if de in [x.link for x in iterc]:
974             print("\033[91mSelf in\033[00m", self)
975
976         for cable in self.cables:
977             if cable.fromobj == self or cable.toobj == self:
978                 cable.delete()
979                 break
980
981         de.update()
982
983         if self.__class__.__name__ == "Switch":
984             self.disconnectport(de)
985         elif de.__class__.__name__ == "Switch":
986             de.disconnectport(self)
987
988     self.update()
989
990 def delete(self, *widget, conf=1, pr=1):
991     if pr == 1:
992         yonW = YesOrNoWindow("¿Estás seguro de que quieres eliminar " + self.name + " definitivamente? El  

993             ↪ objeto será imposible de recuperar y te hechará de menos.")
994         yonR = yonW.run()
995         yonW.destroy()
996     else:
997         yonR = 1
998     if yonR == 1:
999         self.disconnect(0, de="All")
1000         objlst.delete(self)
1001         self.image.destroy()
1002         global allobjects
1003         allobjects.remove(self)
1004     elif yonR == 0:
1005         print("Piénsatelo dos veces")
1006     else:
1007         raise
1008
1009 def packet_received(self, pck, *args, port=None):
1010     print("Hola, soy {} y he recibido un paquete, pero no sé que hacer con él".format(self.name))
1011     if config.getboolean("DEBUG", "packet-received"):
1012         print(">Pck:", pck)
1013         if pck.frame != None:
1014             print("\033[91m>>Atributos del paquete\033[00m")
1015             totalen = pck.lenght + 14*8
1016             wfr = bformat(pck.frame, (totalen+14)*8)
1017             print(">Wfr:", wfr)
1018             mac1 = "{0:011b}".format(pck.frame)[0:6*8]

```



```

1018         print(">Mac:", int(mac1,2))
1019         readmac = str(hex(int(mac1,2))).strip("0x")
1020         print(":".join([readmac[i * 2:i * 2 + 2] for i,bl in enumerate(readmac[::2])]).upper())
1021
1022         print("<<Fin de los atributos")
1023
1024     class mac():
1025         def __init__(self, *macaddr, bits=48):
1026             print("macaddr:", *macaddr)
1027             if macaddr == None or True:
1028                 tmp = self.genmac(bits=bits)
1029
1030             self.int = tmp[0]
1031             self.str = tmp[1]
1032             self.bin = ("{:0}" + str(bits) + "b").format(self.int)
1033
1034         def genmac(*self, bits=48, mode=None):
1035             #Por defecto se usa mac 48, o lo que es lo mismo, la de toa la vida
1036             #Nota, falta un comprobador de que la mac no se repita
1037             realmac = int("11" + str("{:0}" + str(bits-2) + "b").format(random.getrandbits(bits-2)),2)
1038             readmac = str(hex(realmac)).upper().replace("0X", "")
1039             readmac = ":".join([readmac[i * 2:i * 2 + 2] for i,bl in enumerate(readmac[::2])])
1040             if mode == 0:
1041                 return realmac
1042             if mode == 1:
1043                 return readmac
1044             else:
1045                 return [realmac, readmac]
1046
1047         def __str__(self):
1048             readmac = str(hex(self.int)).upper().replace("0X", "")
1049             return ":".join([readmac[i * 2:i * 2 + 2] for i,bl in enumerate(readmac[::2])])
1050
1051         def __bytes__(self):
1052             return Object.__bytes__(self)
1053
1054         def __int__(self):
1055             return self.int
1056         def __index__(self):
1057             return self.int
1058         def list(self):
1059             return self.str.split(":")
1060
1061     npack = 0
1062
1063     class Router(ObjectBase):
1064         cnt = 1
1065         def __init__(self, x, y, *args, name="Default"):
1066             global cnt_objects
1067             self.objecttype = "Router"
1068             push_elemento("Creado Objeto Router")
1069
1070             ObjectBase.__init__(self, x, y, self.objecttype, name=name)
1071             self.x = x
1072             self.y = y
1073
1074         def __del__(self, *args):
1075             push_elemento("Eliminado objeto")
1076             del self
1077
1078     ### ESTO ERA NESTED DE SWITHC ###
1079
1080     class Port():
1081         def __init__(self, switch):
1082             self.id = switch.portid
1083             self.dic = switch.pdic
1084             self.all = switch.pall
1085             switch.portid += 1
1086             self.switch = switch
1087             self.connection = None

```

```

1088         self.all[self.id] = self
1089         self.dic[self.id] = self.connection
1090     def connect(self, connection):
1091         self.connection = connection
1092         self.dic[self.id] = self.connection
1093     def disconnect(self):
1094         self.connection = None
1095         self.dic[self.id] = self.connection
1096     def is_available(self):
1097         if self.connection == None:
1098             return True
1099         return False
1100
1101 class w_switch_table(Gtk.ApplicationWindow):
1102     def __init__(self, switch):
1103         self.link = switch
1104         builder = switch.builder
1105         builder.get_object("window_switch-table-button").connect("clicked", self.hide)
1106         builder.get_object("window_switch-table").connect("delete-event", self.hide)
1107         self.store = Gtk.ListStore(str,int,int,int)
1108
1109         self.view = builder.get_object("window_switch-table-TreeView")
1110         self.view.set_model(self.store)
1111         for i, column_title in enumerate(["MAC", "Puerto", "TTL (s)"]):
1112             renderer = Gtk.CellRendererText()
1113             column = Gtk.TreeViewColumn(column_title, renderer, text=i)
1114             column.set_sort_column_id(i)
1115             self.view.append_column(column)
1116         self.ticking = False
1117         builder.get_object("window_switch-table").set_keep_above(True)
1118
1119     def show(self, *a):
1120         self.ticking = True
1121         GObject.timeout_add(1001, self.tick)
1122         for row in self.store:
1123             row[2] = row[3] - time.time()
1124         self.link.builder.get_object("window_switch-table").show_all()
1125
1126     def hide(self, window, *event):
1127         self.link.builder.get_object("window_switch-table").hide()
1128         self.ticking = False
1129         return True
1130
1131     def append(self, lst):
1132         lst.append(lst[2])
1133         for row in self.store:
1134             row[2] = row[3] - time.time()
1135         print(lst)
1136         row = self.store.append(lst)
1137         print(self.view.get_property("visible"))
1138         if self.view.get_property("visible") == True:
1139             self.ticking = True
1140             GObject.timeout_add(1001, self.tick)
1141
1142     def tick(self):
1143         for row in self.store:
1144             row[2] = row[3] - time.time()
1145             if row[2] <= 0:
1146                 try:
1147                     self.store.remove(row.iter)
1148                     self.link.table.remove(row)
1149                 except:
1150                     pass
1151             if len(self.store) == 0:
1152                 self.ticking = False
1153                 return self.ticking
1154
1155     def remove(self, lst):
1156         for row in self.store:
1157             if row == lst:
1158                 self.store.remove(row.iter)
1159                 self.link.table

```

```

1158         break
1159     pass
1160
1161 class Switch(ObjetoBase):
1162     cnt = 1
1163     #El objeto puerto
1164
1165     def __init__(self, x, y, *args, name="Default", maxconnections=5):
1166         self.objectype = "Switch"
1167         self.portid = 0
1168         self.pdic = {}
1169         self.pall = {}
1170
1171         push_elemento("Creado objeto Switch")
1172         self.imgdir = resdir + "Switch.*"
1173         ObjetoBase.__init__(self, x, y, self.objectype, name=name, maxconnections=maxconnections)
1174         self.x = x
1175         self.y = y
1176         self.timeout = 20 #Segundos
1177
1178         for p in range(self.max_connections):
1179             Port(self)
1180         print(self.pall)
1181
1182         self.table = [
1183             #[MAC, port, expiration]
1184         ]
1185         self.wtable = w_switch_table(self)
1186         child = Gtk.MenuItem.new_with_label("Routing Table")
1187         self.builder.get_object("grid_rclick").append(child)
1188         child.connect("activate", self.wtable.show)
1189         child.show()
1190
1191         self.ch = child
1192
1193     def load(self):
1194         ObjetoBase.load(self)
1195         del self.wtable
1196         self.table = []
1197         self.wtable = w_switch_table(self)
1198
1199         del self.ch
1200         child = Gtk.MenuItem.new_with_label("Routing Table")
1201         self.builder.get_object("grid_rclick").append(child)
1202         child.connect("activate", self.wtable.show)
1203         child.show()
1204
1205         self.ch = child
1206
1207
1208     def connectport(self, objeto):
1209         for port in self.pall:
1210             if self.pall[port].is_available():
1211                 self.pall[port].connect(objeto)
1212                 break
1213         print(self.pdic)
1214
1215     def disconnectport(self, objeto):
1216         for p in self.pdic:
1217             print("i: {}, idx: {}".format(p, self.pdic[p]))
1218             if objeto == self.pdic[p]:
1219                 self.pall[p].disconnect()
1220                 break
1221         print(self.pdic)
1222
1223     def packet_received(self, pck, port=None):
1224         macd = "{0:0112b}".format(pck.frame)[0:6*8]
1225         macs = "{0:0112b}".format(pck.frame)[6*8+1:6*16+1]
1226
1227         #LO PRIMERO: AÑADIRLO A LA TABLA

```

```

1228 readmac = str(hex(int(macs,2))).upper().replace("0X", "")
1229 readmac = ":".join([readmac[i * 2:i * 2 + 2] for i,bl in enumerate(readmac[:2])])
1230
1231 for tab in self.table:
1232     if tab[2] <= time.time():
1233         print("Ha llegado tu hora")
1234         self.table.remove(tab)
1235         self.wtable.remove(tab)
1236     if tab[0] == int(macd,2):
1237         print("TAB[0] == mcd")
1238         tab[2] = int(time.time()+self.timeout)
1239         for row in self.wtable.store:
1240             print(row[0], tab[0])
1241             if int(row[0].replace(":", ""),16) == tab[0]:
1242                 row[3] = int(time.time()+self.timeout)
1243 if int(macs,2) not in [x[0] for x in self.table]:
1244     tmp = [int(macs,2), port, int(time.time()+self.timeout)]
1245     self.table.append(tmp)
1246     tmp = [readmac, port, int(time.time()+self.timeout)]
1247     self.wtable.append(tmp)
1248
1249 #####
1250
1251 #ObjetoBase.packet_received(self, pck)
1252
1253 ttl = int(pck.str[64:72],2)
1254 ttlnew = "{0:08b}".format(ttl-1)
1255 pck.str = "".join(( pck.str[:64], ttlnew, pck.str[72:] ))
1256
1257 print("self.macdir",int(self.macdir), int("{0:0112b}".format(pck.frame)[6*8+1:6*16+1],2))
1258 print("TTL:", int(pck.str[64:72],2), pck.str[64:72])
1259
1260 print("Soy {} y mi deber es entregar el paquete a {}".format(self.name,int(macd,2)))
1261 print("El paquete llegó por el puerto {}".format(port))
1262 dic = {}
1263 for i in self.connections:
1264     dic[int(i.macdir)] = i
1265 print("Connections MAC's:", dic)
1266
1267 #Cambiamos los bits de macs
1268 #Si macd en conn, enviarle el paquete
1269 #Si existe una tabla de enrutamiento que contiene una ruta para macd, enviar por ahi
1270 #Si no, enviar al siguiente, y así
1271 print(">MAAAC:",int(macd,2), "DIIIC:")
1272 if int(macd,2) in dic and ttl > 0:
1273     pck.animate(self, dic[int(macd,2)])
1274
1275 elif int(macd,2) in [x[0] for x in self.table] and ttl >= 0:
1276     for x in self.table:
1277         if x[0] == int(macd,2):
1278             pck.animate(self, self.pdic[x[1]])
1279
1280 elif "Switch" in [x.objectype for x in self.connections] and ttl >= 0:
1281     print("Ahora lo enviamos al siguiente router")
1282     print(int(macd,2), dic)
1283     tmp1st = self.connections[:] #Crea una nueva copia de la lista
1284     print(tmp1st)
1285     for i in tmp1st:
1286         if int(macs,2) == int(i.macdir):
1287             print("REMOVING", i)
1288             tmp1st.remove(i)
1289     try:
1290         tmp1st.remove(*[x for x in tmp1st if x.objectype == "Computer"])
1291     except TypeError:
1292         pass
1293     print("Tmp1st:", tmp1st)
1294     obj = choice(tmp1st)
1295     print("Sending to:", obj)
1296     pck.animate(self, obj)
1297

```

```

1298     def debug(self, *args):
1299         print(self.pdic)
1300         print("MyMac:", self.macdir)
1301         row_format = "{:>20}" * 3
1302         print(row_format.format("MAC", "NXT", "EXP s"))
1303         for row in self.table:
1304             if row[1] == None:
1305                 row[1] = "None"
1306             if int(row[2]-time.time()) <= 0:
1307                 self.table.remove(row)
1308             print(row_format.format(row[0], row[1], int(row[2]-int(time.time()))))
1309
1310     #¿Tengo permisos de escritura?, no se si tendré permisos
1311     #Update: Si los tenía
1312     class Hub(ObjetoBase):
1313         cnt = 1
1314         def __init__(self, x, y, *args, name="Default", maxconnections=4, ip=None):
1315             self.objecttype = "Hub"
1316             push_elemento("Creado objeto Hub")
1317             self.imgdir = resdir + "Hub.*"
1318             ObjetoBase.__init__(self, x, y, self.objecttype, name=name)
1319             self.x = x
1320             self.y = y
1321
1322         def packet_received(self, pck, port=None):
1323             ttl = int(pck.str[64:72], 2)
1324             macs = "{0:0112b}".format(pck.frame)[6*8+1:6*16+1]
1325             ttlnew = "{0:08b}".format(ttl-1)
1326             pck.str = "".join(( pck.str[:64], ttlnew, pck.str[72:] ))
1327             if ttl >= 0:
1328                 for obj in self.connections:
1329                     pck.animate(self, obj)
1330
1331     class Computador(ObjetoBase):
1332         cnt = 1
1333         def __init__(self, x, y, *args, name="Default", maxconnections=1, ip=None):
1334             self.objecttype = "Computer"
1335
1336             push_elemento("Creado objeto Computador")
1337             self.img = resdir + "Comp.*"
1338             ObjetoBase.__init__(self, x, y, self.objecttype, name=name)
1339             self.x = x
1340             self.y = y
1341             self.max_connections = maxconnections
1342             self.IP = None
1343
1344             self.pingwin = PingWin(self)
1345             self.builder.get_object("grid_rclick-sendpkg").connect("activate", self.pingwin.show)
1346
1347             self.update()
1348
1349         def load(self):
1350             ObjetoBase.load(self)
1351             self.pingwin = PingWin(self)
1352             self.builder.get_object("grid_rclick-sendpkg").connect("activate", self.pingwin.show)
1353
1354     class ip():
1355         def __init__(self, *args, ipstr="None"):
1356             self.str = ipstr
1357
1358         def __str__(self):
1359             return self.str
1360
1361         def set_str(self, str):
1362             self.str = str
1363             self.parser(str, 0)
1364
1365         def set_bin(self, binar):
1366             t = binar
1367             print(bin(t))

```

```

1368         if "0b" not in str(t) and "." in str(t):
1369             print("Type is str")
1370             self.bins = t
1371         elif "0b" in str(bin(t)) and "." not in str(bin(t)):
1372             print("Type is binar")
1373             self.bin = t
1374         else:
1375             print("Error:", t)
1376         self.parser(t, 1)
1377
1378     #ip2p stands 4 'ip to parse'
1379     def parser(self, ip2p, mode):
1380         #mode 0: str2b
1381         if mode == 0:
1382             tmp1st = ip2p.split(".")
1383             toreturn = []
1384             for i in tmp1st:
1385                 i = int(i)
1386                 toreturn.append("{0:08b}".format(i))
1387             self.bins = ".".join(toreturn)
1388             self.bin = int(self.bins.replace(".", ""), base=2)
1389             return self.bins
1390
1391         #mode 1: b2str
1392         elif mode == 1:
1393             if "0b" not in str(ip2p):
1394                 self.bin = bin(int(ip2p.replace(".", ""), base=2))
1395                 self.str = ".".join([str(int(i, base=2)) for i in ip2p.split(".")])
1396             elif "0b" in str(ip2p):
1397                 print("La ip", ip2p, "es bin")
1398                 tmp = str(ip2p).replace("0b", "")
1399                 n = 8
1400                 self.bins = ".".join([tmp[i * n:i * n + n] for i, blah in enumerate(tmp[::n])])
1401                 self.str = ".".join([str(int(tmp[i * n:i * n + n], base=2)) for i, blah in enumerate(tmp[::n])])
1402             else:
1403                 raise
1404         else:
1405             print("Debug:", mode)
1406             raise NameError('No mode defined')
1407
1408     def update(self):
1409         ObjetoBase.update(self)
1410         self.image.set_tooltip_text(self.name + " (" + str(len(self.connections)) + "/" +
1411             ↳ str(self.max_connections) + ")\n" + str(self.IP))
1412         submenu1 = self.builder.get_object("grid_rclick-sendpkg").get_submenu()
1413         print("Compcon: ", [x.name for x in self.compcon()])
1414
1415         if self.IP != None:
1416             objlst.update(self, "IP", str(self.IP))
1417
1418     #Ahora es cuando viene la parte de haber estudiado.
1419     #SÓLO ENVÍA PINGS, (ICMP)
1420     sub_N = 0
1421     def send_pck(self, *widget, to=None):
1422         global npack
1423         Sub_N = Computador.sub_N
1424         #nonlocal sub_N
1425         de = self
1426         print(widget)
1427         if to == None:
1428             to = widget[0].link
1429
1430         print("fnc send_pck from {} to {}".format(self.name, to.name))
1431
1432         if MainClase.has_ip(self) and MainClase.has_ip(to):
1433             print("Continuando")
1434         else:
1435             print("Un objeto no tiene IP")
1436             yonW = YesOrNoWindow("Uno o los dos objetos no tienen dirección IP", Yest="OK", Not="Ok también")
1437             yonR = yonW.run()

```

```

1437         yonW.destroy()
1438         raise Exception("Un objeto no tiene IP")
1439     #Ambos deben tener direccion ip
1440     #def __init__(self, header, payload, trailer, cabel=None):
1441     ping = Ping.create(0, self.IP, to.IP)
1442     Sub_N += 1
1443     npack += 1
1444
1445     print("PCK ICMP HEADER:", "{0:064b}".format(ping.icmp_header))
1446     print("PCK IPHEADER:", "{0:0160b}".format(ping.ip_header))
1447
1448     print("MAC's:", self.maddir, to.maddir)
1449     frame = eth(int(to.maddir), int(self.maddir), ping)
1450     frame.applytopack(ping)
1451     print("Pck frame:", ping.frame)
1452
1453     ping.animate(self, self.connections[0])
1454
1455     msg = "{ } >Enviado ping a { }".format(time.strftime("%H:%M:%S"), str(to.IP))
1456     self.pingwin.statusbar.push(self.pingwin.statusbar.get_context_id(msg), msg)
1457
1458     #Ver routing: https://en.wikipedia.org/wiki/IP_forwarding
1459     def packet_received(self, pck, *args, port=None):
1460         print("Hola, soy { } y he recibido un paquete, tal vez tenga que responder".format(self.name))
1461         #Si el tipo de ping es x, responder, si es y imprimir info
1462         if config.getboolean("DEBUG", "packet-received"):
1463             print(">Pck:", pck)
1464             if pck.frame != None:
1465                 frame="{0:0111b}".format(pck.frame)
1466                 print("\033[91m>>Atributos del paquete\033[00m")
1467                 totalen = pck.lenght + 14*8
1468                 print("Frame:", bin(pck.frame))
1469                 mac1 = "{0:0111b}".format(pck.frame)[0:6*8]
1470                 readmac = str(hex(int(mac1,2))).strip("0x")
1471                 print(">Mac1:", " ".join([readmac[i * 2:i * 2 + 2] for i,bl in enumerate(readmac[::2])]).upper())
1472                 readmac = str(hex(int( "{0:0111b}".format(pck.frame)[6*8+1:6*16+1] ,2))).strip("0x")
1473                 print(">Mac2:", " ".join([readmac[i * 2:i * 2 + 2] for i,bl in enumerate(readmac[::2])]).upper())
1474                 print("EtherType:", int(frame[12*8+1:8*14+1],2))
1475                 print("Resto==Bits:", int(frame[8*14+1::],2)==pck.bits)
1476                 print(pck.str)
1477
1478                 n, tmp = 8, pck.str[96:128]
1479                 print("IPs:", " ".join([str(int(tmp[i * n:i * n+n], base=2)) for i,blah in enumerate(tmp[::n])]))
1480                 tmp = pck.str[128:160]
1481                 print("IPd:", " ".join([str(int(tmp[i * n:i * n+n], base=2)) for i,blah in enumerate(tmp[::n])]))
1482
1483                 print("<<Fin de los atributos")
1484             n,tmp = 8, pck.str[128:160]
1485             tmp = pck.str[128:160]
1486             print(int(tmp,2), int(self.IP))
1487             if int(tmp,2) == int(self.IP):
1488                 ty = int("{0:064b}".format(pck.icmp_header)[:8],2)
1489                 if ty == 8:
1490                     print("El paquete era para mí, voy a responder un gracias :D")
1491                     ping = Ping.create(1, self.IP, int(pck.str[96:128],2))
1492                     frame = eth(int("{0:0111b}".format(pck.frame)[6*8+1:6*16+1],2), int(self.maddir), ping)
1493                     frame.applytopack(ping)
1494
1495                     ping.animate(self, self.connections[0])
1496                 elif ty == 0:
1497                     print("De nada")
1498                 else:
1499                     print("ty es:", ty)
1500
1501                 msg = "{ } >Recibido ping de { }".format(time.strftime("%H:%M:%S"), " ".join([str(int(tmp[i * n:i * n+n],
1502                                     ↪ base=2)) for i,blah in enumerate(tmp[::n])]))
1503                 self.pingwin.statusbar.push(self.pingwin.statusbar.get_context_id(msg), msg)
1504
1505 class Servidor(Computador):
1506     def __init__(self, x, y, *args, name="Default", maxconnections=1, ip=None):

```

```

1506         self.objecttype = "Servidor"
1507
1508         push_elemento("Creado objeto {}".format(self.objecttype))
1509         self.img = resdir + "Server.*"
1510         ObjetoBase.__init__(self, x, y, self.objecttype, name=name)
1511         self.x = x
1512         self.y = y
1513         self.max_connections = maxconnections
1514         self.IP = self.ip()
1515
1516     #La clase para los objetos cable
1517     class Cable():
1518         def __init__(self, fromo, to, *color):
1519             lprint("Argumentos sobrantes: ", *color)
1520             self.objecttype = "Wire"
1521             self.fromobj = fromo
1522             self.toobj = to
1523             self.fromx = TheGrid.gridparser(fromo.x, TheGrid.wres,1)
1524             self.fromy = TheGrid.gridparser(fromo.y, TheGrid.hres,1)
1525             self.tox = TheGrid.gridparser(to.x, TheGrid.wres,1)
1526             self.toy = TheGrid.gridparser(to.y, TheGrid.hres,1)
1527             self.w = max(abs(fromo.realx - to.realx),3)
1528             self.h = max(abs(fromo.realy - to.realy),3)
1529
1530             self.cair()
1531
1532             self.x, self.y = min(fromo.x, to.x)-0.5, min(fromo.y, to.y)-0.5
1533
1534             TheGrid.moveto(self.image, self.x, self.y, layout=TheGrid.cables_lay)
1535             lprint("Puesto cable en: ", self.x, "; ", self.y)
1536
1537             self.image.show()
1538
1539             global cables
1540             cables.append(self)
1541             lprint("Todos los cables: ", cables)
1542
1543         def load(self):
1544             global cables
1545             self.cair()
1546             self.image.show()
1547             cables.append(self)
1548
1549             self.fromobj.connect(self.toobj, self)
1550
1551         def cair(self):
1552             fromo = self.fromobj
1553             to = self.toobj
1554             width, height = max(abs(self.fromobj.realx - self.toobj.realx),3), max(abs(self.fromobj.realy -
1555                 ↪ self.toobj.realy),3)
1556             surface = cairo.ImageSurface(cairo.FORMAT_ARGB32, width, height)
1557             ctx = cairo.Context(surface)
1558
1559             #ctx.scale(width, height)
1560
1561             ctx.close_path ()
1562
1563             if config.getboolean("DEBUG", "show-cable-rectangle"):
1564                 ctx.set_source_rgba(0, 0, 1, 0.1) # Solid color
1565                 ctx.rectangle(0,0,width,height)
1566                 ctx.fill()
1567
1568             ctx.set_line_width(1.5)
1569             ctx.set_source_rgb(1,0,0)
1570             if (fromo.x < to.x and fromo.y < to.y) or (fromo.x > to.x and fromo.y > to.y):
1571                 ctx.move_to(0, 0)
1572                 ctx.line_to(width, height)
1573             elif fromo.x == to.x:
1574                 ctx.move_to(width/2, 0)

```



```

1575         ctx.line_to(width/2, height)
1576     elif fromo.y == to.y:
1577         ctx.move_to(0, height/2)
1578         ctx.line_to(width, height/2)
1579     else:
1580         ctx.move_to(0, height)
1581         ctx.line_to(width, 0)
1582
1583     ctx.stroke()
1584
1585     self.image = gtk.Image.new_from_surface(surface)
1586     self.x, self.y = min(fromo.x, to.x)-0.5, min(fromo.y, to.y)-0.5
1587
1588     TheGrid.moveto(self.image, self.x, self.y, layout=TheGrid.cables_lay)
1589
1590     def delete(self):
1591         global cables
1592         cables.remove(self)
1593
1594         self.fromobj.cables.remove(self)
1595         self.toobj.cables.remove(self)
1596
1597         self.image.hide()
1598         print("\033[96mCable\033[00m", self, "\033[96mdeleted\033[00m")
1599         del self
1600
1601     save.classes = [ObjetoBase, Switch, Hub, Computador, Servidor, Cable]
1602
1603     #De momento sólo soportará el protocolo IPv4
1604     class packet():
1605         def __init__(self, header, trailer, payload, cabel=None):
1606             lprint("Creado paquete de res")
1607             self.header = header
1608             self.payload = payload
1609             self.trailer = trailer
1610             #self.packet = header + payload + trailer
1611
1612         def new_from_total(self, bits):
1613             print("Length (bits):", int(bin(bits)[18:33],2)*8)
1614             print("Real length:", int(len(bin(bits))-2 ))
1615             self.bits = bits
1616             self.lenght = int(bin(bits)[18:33],2)
1617             self.str = str("{0:0}" + str(int(int(bin(bits)[18:33],2)*8 )) + "b").format(self.bits)
1618             print(self.str)
1619
1620         def send(self, de):
1621             ##SIN TERMINAR##
1622             ##FALTA AÑADIR TODO LO DEL FRAME##
1623             if de.objecttype == "Computador":
1624                 to = de.connections[1]
1625                 self.animate(de, to)
1626
1627     #Siendo t=fps/s, v=px/s, v default = 84
1628     def animate(self, start, end, fps=30, v=200, color=None, port=None):
1629         if color == None:
1630             if self.color != None:
1631                 color = self.color
1632             else:
1633                 color = "#673AB7"
1634         from math import sqrt, pi
1635         #Long del cable
1636         try:
1637             cable = start.cables[[x.toobj for x in start.cables].index(end)]
1638         except ValueError:
1639             cable = start.cables[[x.fromobj for x in start.cables].index(end)]
1640         w, h = cable.w + TheGrid.sqres, cable.h + TheGrid.sqres
1641         x, y = cable.x*TheGrid.sqres-TheGrid.sqres/2, cable.y*TheGrid.sqres-TheGrid.sqres/2
1642         xi, yi = (start.x-0.5)*TheGrid.sqres-x, (start.y-0.5)*TheGrid.sqres-y
1643         xf, yf = end.x, end.y
1644         r = sqrt(cable.w**2+cable.h**2) #Píxeles totales

```

```

1645     t=r/v #Tiempo en segundos que durara la animacion
1646     tf = int(fps*t) #Fotogramas totales
1647     spf = 1/fps #Segundos por fotograma
1648
1649     sq = 12
1650     surface = cairo.ImageSurface(cairo.FORMAT_ARGB32, w, h)
1651     ctx = cairo.Context(surface)
1652     ctx.close_path()
1653     ctx.set_source_rgba(0,1,1,1)
1654     ctx.arc(-sq/2,-sq/2,sq/2,0,2*pi)
1655     ctx.fill()
1656     ctx.stroke()
1657     ctx.close_path()
1658
1659     image = gtk.Image.new_from_surface(surface)
1660     TheGrid.animat_lay.put(image,x,y)
1661     TheGrid.animat_lay.show_all()
1662
1663     #print("x: {}, y: {}, tf:{}, spf*m:{}, t: {}".format(x/TheGrid.sqres,y/TheGrid.sqres,tf,int(spf*1000), t))
1664     f = 0
1665     x,y = xi,yi
1666     sx,sy = (w-TheGrid.sqres)/tf,(h-TheGrid.sqres)/tf
1667     if start.x > end.x:
1668         sx = -sx
1669     if start.y > end.y:
1670         sy = -sy
1671
1672     def iteration():
1673         nonlocal f
1674         nonlocal x
1675         nonlocal y
1676         nonlocal ctx
1677         nonlocal surface
1678         nonlocal port
1679         if f <= tf:
1680             #Do things
1681             #print("Current f: {}; x,y: {}, {}".format(f, x,y))
1682             x += sx
1683             y += sy
1684
1685             del ctx
1686             surface = cairo.ImageSurface(cairo.FORMAT_ARGB32, w, h)
1687             ctx=cairo.Context(surface)
1688             ctx.set_source_rgba(*hex_to_rgba(color))
1689             ctx.arc(x,y,sq/2,0,2*pi)
1690             ctx.fill()
1691             image.set_from_surface(surface)
1692
1693             f += 1
1694             return True
1695         else:
1696             del ctx
1697             image.destroy()
1698             del surface
1699             #print("Paquete enviado a {}".format(end))
1700             if end.__class__.__name__ == "Switch":
1701                 for p in end.pall:
1702                     if end.pall[p].connection == start:
1703                         port = p
1704                         break
1705                 print("PORT:", port)
1706                 end.packet_received(self,port=port)
1707                 return False
1708             end.packet_received(self, port=port)
1709             return False
1710
1711     GObject.timeout_add(spf*1000, iteration)
1712
1713
1714     return True

```

```

1715
1716     def __str__(self):
1717         return "<" + str(packet) + ">"
1718
1719 # ETHERNET LAYER #
1720 #Usando DIX, más comun en IP
1721 #Al ser emulado no es necesario CRC Checksum
1722 #SIEMPRE 112 longitud (48*2+16)
1723 class eth(packet):
1724     #Se crea el header
1725     def __init__(self, destmac, sourcemac, *pack, EtherType=0x0800):
1726         def corrector(mac):
1727             if type(mac) == str:
1728                 mac2 = 0
1729                 for x in mac.split(":"):
1730                     mac2 = mac2 << 8 | int(x, 16)
1731                 return mac2
1732             elif type(mac) == int:
1733                 return mac
1734             else:
1735                 raise Exception("MAC ERROR")
1736
1737         destmac = corrector(destmac)
1738         sourcemac = corrector(sourcemac)
1739         print("Destmac", "{0:048b}".format(destmac))
1740
1741         self.machheader = (destmac << (6*8+1) | sourcemac) << 16 | EtherType
1742         print(int("{0:0111b}".format(self.machheader)[0:6*8],2))
1743
1744     #Se le añade la payload al frame
1745     def applytopack(self, pack):
1746         self.pack = pack
1747         print(">Mach:", bin(self.machheader).replace("0b", ""))
1748         print(">Pck:", pack)
1749         print(pack.lenght)
1750         ret = (self.machheader << pack.lenght*8) | pack.bits
1751         pack.frame = ret
1752         pack.framesrt = None
1753         print("pack.len: {}, bits len: {}".format(pack.lenght*8, len(bin(pack.bits).strip("0b"))))
1754         print(">Ret:", bin(ret).replace("0b", ""))
1755         print(int("{0:0111b}".format(self.machheader)[0:6*8],2))
1756         return ret
1757
1758     def __str__(self):
1759         return str( bin(self.machheader) )
1760
1761 #Internet Layer
1762 class icmp(packet):
1763     def __init__(self, ipheader, icmpheader, payload):
1764         print("Len:", int(bin(ipheader)[18:33],2)-28)
1765         self.bits = (ipheader << 8*8 | icmpheader) << ( (int(bin(ipheader)[18:33],2) -28) * 8) | payload #BITS
1766         ↪ 16a31 - 28
1767         packet.new_from_total(self, self.bits)
1768
1769     def __str__(self):
1770         return self.str
1771
1772 ### Application layer ###
1773
1774 #Estos paquetes pueden ser Request o Reply.
1775 #El header es de 20 bytes, la payload es de 8 + datos opcionales, pero el estándar es 64 bits.
1776 #Tipo de mensaje es 8 para request y 0 para reply. El ICMP es siempre 0.
1777 class Ping(icmp):
1778     identifi = 0
1779     def __init__(self):
1780         pass
1781
1782     def create(r, sourceip, desti_ip, *n, payload=int( 4.3*10**19 ) << 6 | 42, \
1783         flags=0b010, ttl=32):

```

```

1784         self = Ping()
1785         if r == 0:
1786             Type = 8
1787             self.color = "#4CAF50"
1788         if r == 1:
1789             Type = 0
1790             self.color = "#F44336"
1791
1792         self.payload = payload
1793
1794         vihlto = 0b0100010100000000
1795         #20 Iphheader + 8 ICMPHeader + Payload
1796         lenght = int( 20 + 8 + ( int(math.log(payload, 2))+1)/8 ) #In Bytes
1797         frag_off = 0b00000000000000
1798         protocol = 1
1799         checksum = 0 #No es necesario porque no hay cables
1800         sourceip = int(sourceip)
1801         desti_ip = int(desti_ip)
1802         identific = Ping.identifi
1803         Ping.identifi += 1
1804
1805         self.ip_header = (((((((((vihlto << 16 | lenght)<<16 | identific) << 3 | flags) << 13 | frag_off) \
1806         << 8 | ttl) << 8 | protocol) << 16 | checksum) << 32 | sourceip) << 32 | desti_ip)
1807
1808         identifier = 1*2**15 + 42 * 2**8 + 42
1809         Code = 0
1810         icmp_header_checksum = random.getrandbits(16)
1811         self.icmp_header = (((((((((Type << 8) | Code)<< 16) | checksum) << 16) | identifier) << 16) | identific)
1812         self.pck = icmp(self.ip_header, self.icmp_header, self.payload)
1813
1814         self.str = self.pck.str
1815         self.lenght = self.pck.lenght
1816         self.bits = self.pck.bits
1817
1818         return self
1819
1820 #Ventana para configurar las variables de Config.ini
1821 #Nota: Por terminar
1822 class cfgWindow(MainClase):#MainClase:
1823     def __init__(self, *args):
1824         push_elemento("Invocada ventana de configuracion")
1825         writeonlog("Has invocado a la GRAN VENTANA DE CONFIGURACION <--- Boss")
1826         self.cfgventana = builder.get_object("cfgwindow")
1827         self.cfgventana.connect("key-press-event", self.on_key_press_event)
1828         self.cfgventana.connect("key-release-event", self.on_key_release_event)
1829         self.cfgventana.connect("delete-event", self.hidewindow)
1830
1831         builder.get_object("button2").connect("clicked", self.save)
1832
1833         self.eraselogs = builder.get_object("eraselogs")
1834         self.eraselogs.connect("clicked", self.borrarlogs)
1835
1836         self.cfgbbtn1 = builder.get_object("checkbutton1")
1837         self.cfgbbtn1.connect("toggled", self.bttntoggled)
1838         if config.getboolean("BOOLEANS", "print-key-pressed") == True:
1839             self.cfgbbtn1.set_active(True)
1840         else:
1841             self.cfgbbtn1.set_active(False)
1842
1843         booleans = {"print-key-pressed": "print-key-pressed"}
1844
1845         #TODO ESTO ES PARA LOS SPINNERS
1846
1847         #Todos los spinbuttons necesarios
1848         self.spinbuttons = [
1849             #[label, cfgsect, cfgkey, rangef, ranget, incrementf, increment],
1850             ["Win del wres", "GRAPHICS", "wres", 450, 1600, 5, 10],
1851             ["Win del hres", "GRAPHICS", "hres", 450, 1600, 5, 10],
1852             ["Wres del grid", "GRAPHICS", "viewport-wres", 20, 100, 1, 5],
1853             ["Hres del grid", "GRAPHICS", "viewport-hres", 15, 100, 1, 5],

```

```

1854         ["Res de los sq", "GRAPHICS", "viewport-sqres", 32, 128, 5, 10],
1855         ["Max logs", "DIRS", "Maxlogs", 3, 1000, 1, 5],
1856     ]
1857     self.createdspinbuttons = []
1858
1859     self.spinnergrid = builder.get_object("graph")
1860
1861     def forspin(spinner):
1862         spinbutton = Gtk.SpinButton.new(None, 0, 0)
1863         tplst = spinner
1864         label = Gtk.Label.new(tmplst[0])
1865
1866         self.spinnergrid.insert_row(1)
1867
1868         #spinbutton.set_digits(0)
1869         spinbutton.set_numeric(True)
1870         spinbutton.set_range(tmplst[3], tmplst[4])
1871         spinbutton.set_increments(tmplst[5], tmplst[6])
1872         spinbutton.set_value(config.getfloat(tmplst[1], tmplst[2]))
1873
1874         #attach(child, left, top, width, height)
1875         self.spinnergrid.attach(label, 0, 1, 1, 1)
1876         self.spinnergrid.attach(spinbutton, 1, 1, 1, 1)
1877
1878         self.createdspinbuttons.append(spinbutton)
1879
1880     for spinner in self.spinbuttons:
1881         forspin(spinner)
1882
1883     #self.cfgventana.show_all()
1884
1885     def show(self, *args):
1886         self.cfgventana.show_all()
1887
1888     def on_key_press_event(self, widget, event):
1889         #global allkeys
1890         MainClase.on_key_press_event(self, widget, event)
1891         if "ESCAPE" in allkeys:
1892             push_elemento("Cerrada ventana de Configuracion")
1893             self.cfgventana.hide()
1894
1895         if ("CONTROL_L" in allkeys) and ("S" in allkeys):
1896             self.save()
1897         lprint(MainClase.on_key_press_event(self, widget, event))
1898
1899     def on_key_release_event(self, widget, event):
1900         MainClase.on_key_release_event(self, widget, event)
1901
1902     def btntoggled(self, *args):
1903         if self.cfgbtttn1.get_active() == True:
1904             push_elemento("print-key-pressed set True")
1905             config.set("BOOLEANS", "print-key-pressed", "True")
1906         if self.cfgbtttn1.get_active() == False:
1907             push_elemento("print-key-pressed set False")
1908             config.set("BOOLEANS", "print-key-pressed", "False")
1909
1910     def borrarlogs(self, *lala):
1911         #prompt = YesOrNoWindow("Seguro que quieres borrar los logs?")
1912         #if prompt.on_button_clicked(0) == True:
1913         push_elemento("Borrando logs")
1914         for the_file in os.listdir("logfiles/"):
1915             file_path = os.path.join("logfiles/", the_file)
1916             try:
1917                 if os.path.isfile(file_path):
1918                     os.unlink(file_path)
1919             except e:
1920                 lprint(e)
1921
1922     def save(self, *args):
1923         #[label, cfgsect, cfgkey, rangef, ranget, incrementf, increment],

```

```

1924         lprint(self.createdspinbuttons)
1925         for i in range(len(self.createdspinbuttons)):
1926             tmplst = self.spinbuttons[i]
1927             config.set(tmplst[1], tmplst[2], int(self.createdspinbuttons[i].get_value()))
1928
1929         push_elemento("Configuracion guardada")
1930         with open(configdir, 'w') as cfgfile:
1931             lprint("Guardando archivo de configuracion")
1932             try:
1933                 config.write(cfgfile)
1934             except:
1935                 lprint("Error al guardar la configuracion")
1936
1937         def hidewindow(self, window, *event):
1938             window.hide()
1939             return True
1940
1941         class w_changethings(): #Recordatorio para mi yo futuro: que cada objeto debe tener una...
1942             #0 tal vez no sea necesario... A la hora de llamar a la función, espera ¿Con quien estoy hablando?
1943             #Nota, ver notas escritas en la mesa
1944             def __init__(self, objeto):
1945                 self.window = objeto.builder.get_object("changethings")
1946                 self.name_entry = objeto.builder.get_object("changethings_name-entry")
1947                 self.imagebutton = objeto.builder.get_object("changethings_imagebutton")
1948                 self.applybutton = objeto.builder.get_object("chg_apply")
1949                 self.applybutton.connect("clicked", self.apply)
1950                 self.cancelbutton = objeto.builder.get_object("chg_cancel")
1951                 self.cancelbutton.connect("clicked", self.cancel)
1952                 self.window.connect("delete-event", self.hidewindow)
1953                 self.window.connect("key-press-event", self.on_key_press_event)
1954                 self.window.connect("key-release-event", self.on_key_release_event)
1955                 objeto.builder.get_object("chg_MAC-regen").connect("clicked", self.regenclicked)
1956                 print(objeto.builder.get_object("chg_MAC-regen").set_image(gtk.Image.new_from_stock("gtk-refresh", 1)))
1957
1958                 self.link = objeto
1959                 self.image = Gtk.Image.new_from_pixbuf(objeto.image.get_pixbuf())
1960
1961                 def filter_ip(entry):
1962                     PingWin.filter_ip(0, entry)
1963
1964                 def filter_numshex(widget):
1965                     text = widget.get_text().strip()
1966                     widget.set_text("".join([i for i in text if i in "0123456789ABCDEFabcdef"]))
1967
1968                 objeto.builder.get_object("changethings_entry-IP").connect("changed", filter_ip)
1969
1970                 for i in ["chg_MAC-entry" + str(x) for x in range(0,5)]:
1971                     objeto.builder.get_object(i).connect("changed", filter_numshex)
1972
1973                 if objeto.objecttype != "Computer":
1974                     objeto.builder.get_object("changethings_box-IP").destroy()
1975                     objeto.builder.get_object("grid_label-IP").destroy()
1976
1977                 #self.applybutton.connect("clicked", self.apply)
1978                 #self.cancelbutton.connect("clicked", self.cancel)
1979
1980             def show(self, *widget):
1981                 print("widget:", self.link)
1982                 self.window.show_all()
1983                 self.imagebutton.set_image(self.image)
1984                 self.name_entry.set_text(self.link.name)
1985                 tmplst = self.link.maddir.list()
1986                 for i in tmplst:
1987                     tmpentry = self.link.builder.get_object("chg_MAC-entry" + str(tmplst.index(i)))
1988                     tmpentry.set_text(i)
1989
1990                 #Hacer que muestre/oculte los campos de "IP"
1991                 if self.link.objecttype == "Computer":
1992                     try:
1993                         self.link.builder.get_object("changethings_entry-IP").set_text(str(self.link.IP))

```

```

1994         except AttributeError: #Cuando no tiene una str definida
1995             raise
1996         pass
1997     except TypeError:
1998         raise
1999         pass
2000     except:
2001         raise
2002 else:
2003     pass
2004
2005 def apply(self, *npi):
2006     #acuerdate tambien de terminar esto
2007     #Nota: Hacer que compruebe nombres de una banlist, por ejemplo "TODOs"
2008     yonR = None
2009     lprint(npi)
2010
2011     self.link.name = self.name_entry.get_text()
2012     lprint([ self.link.builder.get_object(y).get_text() for y in ["chg_MAC-entry" + str(x) for x in
2013         ↪ range(0,6)] ])
2014     self.link.maddir.str = ":".join( [ self.link.builder.get_object(y).get_text() for y in ["chg_MAC-entry" +
2015         ↪ str(x) for x in range(6)] ])
2016     self.link.maddir.int = int(self.link.maddir.str.replace(":", ""), 16)
2017     self.link.maddir.bin = "{0:048b}".format(self.link.maddir.int)
2018     if self.link.objecttype == "Computer":
2019         try:
2020             iptemp = self.link.builder.get_object("changethings_entry-IP").get_text()
2021             if iptemp == "":
2022                 pass
2023             elif self.link.builder.get_object("changethings_entry-IP").tmp == 2:
2024                 self.link.IP = ip_address(iptemp)
2025             else:
2026                 yonW = YesOrNoWindow("{} no es una IP válida, por favor, introduzca una IP
2027                 ↪ válida".format(iptemp), Yest="OK", Not="Ok también")
2028                 yonR = yonW.run()
2029                 yonW.destroy()
2030         except:
2031             print(Exception)
2032             raise
2033
2034     lprint("self.link.name", self.link.name)
2035
2036     #self.link.image.set_tooltip_text(self.link.name + " (" + str(self.link.connections) + "/" +
2037     ↪ str(self.link.max_connections) + ")")
2038     self.link.update()
2039     self.window.hide()
2040     if yonR!=None:
2041         self.show()
2042
2043 def cancel(self, *npi):
2044     lprint(npi)
2045     self.window.hide()
2046
2047 def hidewindow(self, window, *event):
2048     window.hide()
2049     return True
2050
2051 def on_key_press_event(self, widget, event):
2052     #global allkeys
2053     MainClase.on_key_press_event(self,widget,event)
2054     if "ESCAPE" in allkeys:
2055         push_elemento("Cerrada ventana de Configuracion")
2056         self.window.hide()
2057
2058 def on_key_release_event(self, widget, event):
2059     MainClase.on_key_release_event(self, widget, event)
2060
2061 def regenclicked(self, widget):
2062     t = ObjetoBase.mac.genmac()[1].split(":")
2063     for i in t:

```

```

2060         tmpentry = self.link.builder.get_object("chg_MAC-entry" + str(t.index(i)))
2061         tmpentry.set_text(i)
2062         tmpentry.show()
2063
2064 class PingWin(Gtk.ApplicationWindow):
2065     def __init__(self, obj):
2066         self.link = obj
2067         builder = obj.builder
2068         self.win = builder.get_object("PingWin")
2069         self.statusbar = builder.get_object("PingWin_Statusbar")
2070         self.entry = builder.get_object("PingWin_entry")
2071         self.entry.set_placeholder_text("192.168.1.XXX")
2072         self.ping = builder.get_object("PingWin_Button")
2073
2074         self.ping.connect("clicked", self.do_ping)
2075
2076         self.entry.connect("changed", self.filter_ip)
2077         self.win.connect("delete-event", self.destroy)
2078
2079     def filter_ip(self, entry):
2080         if entry.get_text().strip("") == "":
2081             entry.override_background_color(Gtk.StateFlags.NORMAL, Gdk.RGBA(*hex_to_rgba("#E57373")))
2082         else:
2083             entry.tmp = 0
2084             text = entry.get_text().strip()
2085             entry.set_text("".join([i for i in text if i in "0123456789."]))
2086             if max([len(x) for x in entry.get_text().split(".")]) > 3:
2087                 print("IP NO VÁLIDA")
2088                 entry.tmp = 1
2089             try:
2090                 if max([int(x) for x in entry.get_text().split(".") if x != ""]) > 254:
2091                     print("IP NO VÁLIDA")
2092                     entry.tmp = 1
2093             except ValueError:
2094                 pass
2095             except:
2096                 raise
2097             if len([x for x in entry.get_text().split(".") if x != ""]) == 4 and entry.tmp==0:
2098                 print("IP ACABADA")
2099                 entry.tmp = 2
2100                 entry.override_background_color(Gtk.StateFlags.NORMAL, Gdk.RGBA(*hex_to_rgba("#9CCC65")))
2101
2102             if entry.tmp == 1:
2103                 entry.override_background_color(Gtk.StateFlags.NORMAL, Gdk.RGBA(*hex_to_rgba("#E57373")))
2104             elif entry.tmp == 0:
2105                 entry.override_background_color(Gtk.StateFlags.NORMAL, Gdk.RGBA(*hex_to_rgba("#FFA726")))
2106
2107     def do_ping(self, widget):
2108         ip = self.entry.get_text()
2109         if self.entry.tmp == 2:
2110             print(self.link.compcn())
2111             to = None
2112             for x in self.link.compcn():
2113                 if ip == str(x.IP):
2114                     to = x
2115                     print("IP: {} from {} in compcn {}".format(ip, to, self.link.compcn()))
2116                     Computador.send_pck(self.link, to=to)
2117                     break
2118             if to == None:
2119                 yonW = YesOrNoWindow("La IP {} no se ha encontrado".format(ip), Yest="OK", Not="Ok también")
2120                 yonR = yonW.run()
2121                 yonW.destroy()
2122
2123         else:
2124             yonW = YesOrNoWindow("{} no es una IP válida, por favor, introduzca una IP válida".format(ip),
2125                                     ↪ Yest="OK", Not="Ok también")
2126             yonR = yonW.run()
2127             yonW.destroy()
2128
2129     def show(self, widget):

```



```

2129         self.win.show()
2130     def destroy(self, window, event):
2131         window.hide()
2132         return True
2133
2134 class about(Gtk.AboutDialog):
2135     def __init__(self):
2136         self.win = builder.get_object("AboutWindow")
2137         self.win.connect("delete-event", self.destroy)
2138         self.win.connect("response", self.destroy)
2139         self.win.add_credit_section("Tutores", ["Julio Sánchez"])
2140         #self.win.add_credit_section("Contribuidores", [""])
2141         self = self.win
2142     def show(self, *args):
2143         print("Showing")
2144         self.win.show()
2145     def destroy(self, *args):
2146         self.win.hide()
2147         return True
2148
2149
2150 #Esta clase te permitirá deshacer acciones, algún día de un futuro lejano.
2151 class Undo():
2152     def __init__(self):
2153         self.lastactions = []
2154
2155 #Esta la pongo fuera porque lo mismo la necesito en otra clase
2156
2157 def exiting(self, *ahfjah):
2158     global log
2159     savelog()
2160     lprint("End time: " + time.strftime("%H:%M:%S"))
2161     print ("Window closed, exiting program")
2162     Gtk.main_quit()
2163
2164 def restart(*args):
2165     global log
2166     savelog()
2167     lprint("End time: " + time.strftime("%H:%M:%S"))
2168     lprint("Restarting program")
2169     print("\033[92m#####\033[00m")
2170     os.chdir(startcwd)
2171     os.execl(sys.executable, sys.executable, *sys.argv)
2172
2173 def leppard():
2174     lprint("Gunter lieben glauchen globen")
2175
2176 writeonlog("Esto ha llegado al final del codigo al parecer sin errores")
2177 writeonlog("O no")
2178 MainClase()
2179
2180 lprint("Actual time: " + time.strftime("%H:%M:%S"))
2181 lprint("Complete load time: " + str(datetime.now() - startTime))
2182 push_elemento("Parece que esta cosa ha arrancado en tan solo " + str(datetime.now() - startTime))
2183 Gtk.main()
2184
2185 print("\033[92m#####\033[00m")

```

D.2 Modules/logmod.py

```

1  #Tenia ganas de probar como va en Python esto de los modulos
2  import time, configparser, os
3  config      = configparser.RawConfigParser()
4  configdir   = "Config.ini"
5  config.read(configdir)
6
7  log = []
8  def writeonlog(thingtowrite, *otherthingstowrite):
9      global log
10     thingtowrite = time.strftime("%H:%M:%S") + "@" + thingtowrite
11     try:
12         thingtowrite += " | " + str(otherthingstowrite)
13     except:
14         pass
15     log.append(thingtowrite + "\n")
16     #if len(log) > 15:
17     #    saveonlog()
18
19
20 def saveonlog():
21     global log
22     with open(config.get("DIRS", "Log"), "a") as logfile:
23         logfile.writelines(log)
24         log = []
25
26 def createlogfile():
27     if not os.path.exists("logfiles/"):
28         os.makedirs("logfiles/")
29     nlogfiles = int(len(os.listdir("logfiles/")))
30     if nlogfiles >= int(config.get("DIRS", "Maxlogs")):
31         #print(nlogfiles)
32         #print(int(config.get("DIRS", "Maxlogs")) - nlogfiles)
33         #for i in range(abs(nlogfiles - int(config.get("DIRS", "Maxlogs")))):
34         while nlogfiles > int(config.get("DIRS", "Maxlogs")):
35             #Aqui pones que borre el archivo mas viejo
36             nlogfiles -= 1
37             log.append("Borrado: " + str(min(os.listdir("logfiles/")))+ "\n")
38             try:
39                 os.remove("logfiles/" + min(os.listdir("logfiles/")))
40             except OSError:
41                 print("\033[31mError de I/O en {}, borrar la carpeta de
42                 ↪ logfiles\033[00m".format(str(OSError.filename)))
43             except:
44                 raise
45     try:
46         newlogfile = "logfiles/" + time.strftime("%y%m%d%H%M%S") + " " + config.get("DIRS", "Log")
47         try:
48             os.rename("Log.log", newlogfile)
49         except:
50             print('Ojo cuidao que no se ha podido renombrar <Log.log>')
51     except:
52         pass

```

D.3 Modules/save.py

```

1  print("Module save imported")
2  import pickle
3  import gi
4  import gi.repository
5  gi.require_version('Gtk', '3.0')
6  from gi.repository import Gtk, GObject, Gdk, GdkPixbuf
7
8  gladefile = "Interface2.glade"
9  last = 0
10 asgl = 1
11
12 ### AUN NO FUNCIONA ###
13
14 def save(allobjects, cabls, aslc=0):
15     global asgl
16     global last
17     if aslc | asgl:
18         asgl = 0
19         sw = loadWindow(mode=1)
20         fil = sw.run()
21         sw.destroy()
22     else:
23         fil = last
24     if fil != 0:
25         print(fil.split(".")[1])
26         if fil.split(".")[1] != "inv":
27             print("Nombre de archivo {} no tiene extensión .inv".format(fil))
28             fil += ".inv"
29         last = fil
30         try:
31             os.remove(fil)
32         except:
33             pass
34         print(allobjects)
35         with open(fil, "wb") as output:
36             pickle.dump((allobjects, cabls), output)
37
38 def load(allobjects, cabls):
39     lw = loadWindow()
40     fil = lw.run()
41     lw.destroy()
42     print(fil)
43     if fil != 0:
44         global last
45         global asgl
46         asgl = 0
47         last = fil
48         while len(allobjects) > 0:
49             allobjects[0].delete(pr=0)
50         while len(cabls) > 0:
51             cabls[0].delete()
52         with open(fil, "rb") as inpt:
53             allobj, cables = pickle.load(inpt)
54             print(allobj)
55             print(cables)
56             for obj in allobj:
57                 obj.load()
58             for cable in cables:
59                 cable.load()
60
61 class loadWindow(Gtk.Window):
62     def __init__(self, mode=0):
63         self.builder = Gtk.Builder()
64         self.builder.add_from_file(gladefile)
65         self.window = self.builder.get_object("window-filechooser_load")

```

```
66     filt = Gtk.FileFilter.new()
67     filt.add_pattern("*.inv")
68     filt.set_name("Archivos .inv")
69     self.window.add_filter(filt)
70     todos = Gtk.FileFilter.new()
71     todos.add_pattern("*")
72     todos.set_name("Todos los tipos de archivo")
73     self.window.add_filter(todos)
74     if mode == 1:
75         print("Saving")
76         self.window.set_action(Gtk.FileChooserAction.SAVE)
77         self.builder.get_object("window-filechooser_load-this").set_label("Guardar")
78
79     def run(self):
80         rs = self.window.run()
81         self.window.hide()
82         if rs == 1:
83             rs = self.window.get_filename()
84             self.window.destroy()
85         return rs
86     def destroy(self):
87         del self
```

D.4 Config.ini

```
1  [GRAPHICS]
2  wres = 1000
3  hres = 700
4  viewport-wres = 55
5  viewport-hres = 35
6  viewport-sqres = 50
7  toolbutton-size = 25
8  cable-color = red
9  window-set-keep-above = False
10 start-centered = False
11 viewport-background-color = #FFFFFF
12 revealer-show-default = False
13
14 [DIRS]
15 log = Log.log
16 #Carpeta del pack de recursos:Nah
17 respack = resources/Seriouspack/
18 maxlogs = 20
19
20 [BOOLEANS]
21 print-key-pressed = False
22
23 [SWITCH]
24 routing-ttl = 10
25
26 [DEBUG]
27 show-cable-rectangle = False
28 packet-received = False
```

Este documento esta realizado bajo licencia Creative Commons
«Reconocimiento-CompartirIgual 4.0 Internacional».



El software InvProy está realizado bajo la licencia GNU GPLv3



Encontrará una copia del código y la licencia en <https://github.com/daviddavo/InvProy>

Encontrará una copia de este documento en <https://github.com/daviddavo/InvProy-text>

InvProy Copyright © 2016 David Davó Laviña
david@ddavo.me <http://ddavo.me>

Podrá encontrar una copia digital del documento, el software, las licencias y
los recursos usados en el CD adjunto.

