**Arquitectura de Redes 2**

**Entorno de trabajo**

**Introducción**

Los equipos empleados para la realización de estas prácticas son un conjunto de ordenadores personales bajo sistema operativo GNU/Linux, conectadas entre sí mediante una red de área local de Ethernet.

Los elementos necesarios para realizar las prácticas, y que ya se encuentran instalados en los ordenadores del laboratorio, son los siguientes:

* Adaptador Ethernet, para la conexión a la red de área local.
* Soporte básico de comunicaciones a través del servidor **ethd**, que proporciona acceso directo a la tarjeta Ethernet. Este servidor es necesario porque las funciones 'normales' del sistema operativo para manejar la red trabajan a un nivel mucho más alto, equivalente al nivel de transporte, y para manejar la tarjeta de red a bajo nivel (nivel de tramas o nivel MAC) hay que utilizar una interfaz especial, accesible únicamente por un servidor que se ejecuta bajo la autoridad del usuario administrador del sistema (*root*).
* Librería de programación **libnivel1**. Es una librería de enlace dinámico (DLL) que proporciona las funciones básicas de envío y recepción de tramas, además de otras de inicialización y notificación de llegada de trama.
Los prototipos de las funciones de dicha librería se encuentran en el archivo**/usr/local/include/nivel1.h**, y la documentación de las mismas se encuentra en el documento [API libnivel1](https://moodle.uam.es/mod/resource/view.php?id=705284).
* Sistema de desarrollo: compilador de C *gcc*, programa *make*para realizar la comprobación de dependencias y general el código ejecutable*,*depurador *ddd*, y otra serie de utilidades adicionales, como editores, herramientas de empaquetamiento, etc.

La relación y modo de comunicarse de estos programas se puede ver en la siguiente figura:



**Estos programas ya se encuentran instalados en los laboratorios de tercer curso.**

***Si se desea realizar las prácticas en algún ordenador particular***, debe partir de una versión de GNU/Linux similar a la RedHat 7.3 o superior, en la que se encuentren instalados como mínimo los programas descritos como sistema de desarrollo. El servidor ethd y la librería libnivel1 se pueden descargar directamente desde esta página web, del archivo [ethd-4.1.0-bin.tgz](https://moodle.uam.es/mod/resource/view.php?id=705285),
Estos programas están compilados sobre GNU/Linux RedHat 7.3. Su funcionamiento en otras distribuciones de Linux depende de los niveles de librerías del sistema de que se disponga. Las pruebas que se han realizado con los programas indican que tienen una transportabilidad muy alta, y funcionan sin problemas en casi todas las distribuciones posteriores a la indicada, sean o no de RedHat.

Para proceder a su instalación, descargar este archivo, descomprimirlo en un directorio temporal con el comando

tar -xvzf ethd-4.1.0-bin.tgz

y seguir las instrucciones de instalación del fichero README.INSTALL que se debe encontrar en ese directorio.

**Programas de ejemplo**

Para comprobar la correcta instalación de los archivos de soporte del sistema, y realizar un primer entrenamiento con el entorno de trabajo, se proporcionan unos programas de ejemplo. Estos programas son los siguientes:

* app1\_env.c: Programa de ejemplo de envío de tramas.
* app1\_rec.c: Programa de ejemplo para la recepción de tramas.
* rc\_funcs.c y rc\_funcs.h: Funciones auxiliares proporcionadas para facilitar el desarrollo de las prácticas.
* Makefile: Archivo para realizar la compilación y enlace de los programas de ejemplo mediante el comando *make*. Además, sirve de modelo para realizar archivos similares para el resto de las prácticas.

**Estos programas no se encuentran instalados en los laboratorios de tercer curso**.

Los programas de ejemplo se encuentran en el archivo [ethd-ejemplos-4.1.0.tgz](https://moodle.uam.es/mod/resource/view.php?id=705286). que se puede descargar directamente desde esta página. Para proceder a su instalación, descargar este archivo, y descomprimirlo con el comando

tar -xvzf ethd-ejemplos-3.0.1.tgz

Este comando creará un directorio denominado rc1-ejemplos, donde se encontrarán los archivos citados.

**Servidor ethd. Parámetros de arranque.**

Antes de empezar a enviar tramas hay que arrancar el servidor ethd. El arranque de este programa se realiza ejecutando el siguiente comando:

 **ethd [nivel\_trazas [prob\_error [v [archivo\_error] ]]]**

donde:

* **nivel\_trazas**: El servidor ethd tiene posibilidad de realizar trazas de las tramas que circulan a través de él. La realización y el detalle de estas trazas se regula por el valor asignado a esta parámetro en el arranque del servidor.
Cada una de las posibles trazas que realiza el módulo tiene asignado un valor decimal. El número a pasar como parámetro al servidor será la suma de los valores decimales de cada traza que se desee activar. Estos valores decimales son los siguientes:
	+ 1: Traza de una marca de tiempo (*timestamp*) en envío y recepción trama.
	+ 2: Traza de datos completa de las tramas enviadas y recibidas, en hexadecimal y ascii.
	+ 4: Traza generación de errores en las tramas por parte del servidor. Cada vez que el servidor genere un error (ver siguiente parámetro) se imprimirá una traza que lo indique.
	+ 8: Trazas de llamadas de la interfaz de programación de aplicaciones (API). Corresponden con las funciones de nivel 1 que se ejecutan en el programa del alumno (ver ).
	+ 16: Traza con la interpretación de las tramas enviadas conforme a la estructura de las mismas establecidas para el nivel de enlace con el que se trabaja en las prácticas. Dicha estructura se explica en las prácticas 2 y siguientes.
	+ 32: Trazas internas del módulo. Sçolo es útil para depuración de ethd.
	+ 64: Trazas internas de la comunicación entre nivel 1 y ethd, incluyendo paquete completo de datos de intercambio entre ambos programas. Sólo es útil para depuración de ethd.

Por tanto, el primer parámetro de la llamada a ethd deberá ser la suma de todas las trazas que se desea activar. Así, por ejemplo, para logra traza de timestamp de datos, generación de errores e interpretación de tramas a nivel 2 se debe realizar la siguiente llamada:

 ethd 21

ya que: 21 = 1 + 4 + 16.

Se recomienda como mínimo arrancar ethd con el nivel de trazas 7.

* **prob\_error**: Probabilidad, en partes por millón, de que se produzca un error en un bit en la transmisión. El programa ethd simula con ella la generación de errores en el canal de comunicaciones. Si no se especifica este parámetro, la probabilidad de error que se considera es 0 (transmisión libre de errores).
El modelo implementado considera la probabilidad de que se produzca error en un bit es independiente de la del resto de los bits. La probabilidad de transmisión correcta de una trama completa depende de su longitud. La siguiente tabla muestra estas probabilidades para diversas probabilidades de error de bit, y tres longitudes de trama:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Longitud trama** | **10-5** | **2·10-5** | **5·10-5** | **10-4** | **5·10-4** |
| Trama 60 Bytes | 0.9952 | 0.9904 | 0.9763 | 0.9531 | 0.7866 |
| Trama 256 Bytes | 0.9797 | 0.9599 | 0.9027 | 0.8148 | 0.3591 |
| Trama 1514 Bytes | 0.8859 | 0.7849 | 0.5457 | 0.2978 | 0.0023 |
| Tabla 1: Probabilidad de transmisión correcta de una trama en función de su longitud y la probabilidad de error de bit. |

* **v**: Modo de comunicación virtual. En este modo, el servidor ethd permite la comunicación entre dos aplicaciones en el mismo ordenador ejecutándose en sesiones diferentes como si se encontraran en ordenadores distintos.
La utilización del modo virtual es totalmente transparente a los programas cliente, es decir, las prácticas no necesitan ningún tipo de modificación para funcionar en modo virtual y viceversa. Únicamente es necesario añadido una variable de entorno (EXPORT) en cada sesión en la que se vaya a ejecutar una aplicación, que permita identificar al ordenador emulado que representa. Se realiza mediante un comando export, de la forma:

export EHOST=*n*

donde n es el identificador de host virtual asignado. A este host le corresponderá la siguiente dirección virtual:

MAC: 00:00:00:00:00:n

Si desde dos sesiones se especifica el mismo valor a EHOST, el programa ethd devolverá un error al programa cliente.

* **archivo\_error**: Como alternativa a la introducción aleatoria de errores que poduce el parámetro prob\_error, el servidor ethd tiene la capacidad de producir errores conforme a la intepretación a nivel 2 que se realiza de las tramas que circulan por él. Este mecanismo de introducción de errores es muy útil para depurar los protocolos y todas las condiciones que se exigen a los mismos a partir de la práctica 2.
La generación de errores se hace leyendo instrucciones de un archivo externo, cuyo nombre se pasa en el parámetro archivo\_error. Este archivo tiene la siguiente estructura:
	+ Comentarios: Líneas que comiencen por el carácter '#'.
	+ Líneas en blanco, o que comiencen por ' ' se ignoran.
	+ Líneas de instrucciones: tienen la siguiente sintaxis:

 <control> <accion> <# <comentario>>

* + <control> es un texto con un tipo de trama (I, RR, RNR, SABM, DISC, UA)
	+ <accion> un entero:
		- 0: indica aceptar la trama.
		- 1: indica introducir error en esa trama
	+ <# <comentario>>: Comentario a la línea. ethd lo ignora.
	+ Línea de instrucciones especiales de finalización del archivo. La última línea del archivo puede tener los siguientes comandos:
		- FIN: Finalización del proceso. No se introducirán más errores de este tipo. Si se solicitaron errores aleatorios en la llamada a ethd, se seguirán introduciendo errores de acuerdo a ese modo de funcionamiento.
		- RST: Reiniciación del archivo. Se seguirá trabajando co la introducción de errores sistemáticos de acuerdo al contenido del mismo archivo.

Cuando ethd recibe como 4º parámetro el nombre de un archivo que existe, opera de la siguiente manera:

* + Lee el archivo hasta que encuentra una línea que no sea comentario
	+ La almacena internamente como "acción solicitada".
	+ Recibe una solicitud de envío de una trama.
		- Si el tipo de trama no coincide con la de la acción solicitada, la envía sin errores, y mantiene la misma acción solicitada.
		- Si el tipo de trama coincide con la de la acción solicitada, la envía:
			* Si la acción solicitada era 0, sin errores.
			* Si la acción solicitada era 1, con errores.
			* En cualquiera de estos dos casos, busca una nueva acción solicitada en el archivo.

Lo más recomendable es arrancar ethd en una ventana independiente, para ir viendo los mensajes que genera. Para finalizarlo basta con pulsar Ctrl-C en la misma sesión, o enviar una señal SIGTERM al proceso mediante el comando kill, esto es:

kill -TERM  *process\_id* ;

Cualquiera de estos modos de finalización produce un cierre ordenado, con lo que el programa podrá volver a arrancar normalmente. **No utilizar la señal -9 para finalizarlo**. Este tipo de cierre no elimina el archivo temporal

/tmp/ethd\_socket

que utiliza el proceso, con lo que el programa no se podrá volver a ejecutar. Dicho archivo sólo puede ser eliminado manualmente por el usuario *root*.