Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
Curso de Redes I CI-4815

Proyecto 1

Objetivo general del proyecto:
Que los alumnos comprendan y apliquen el esquema de programación cliente/servidor en una aplicación de red.
Adicionalmente se desea que los estudiantes:

- Mejoren sus destrezas de programación en Lenguaje C.
- Adquieran destrezas de programación con las bibliotecas de *sockets* para la comunicación entre procesos.
- Comprendan el funcionamiento general de los protocolos de comunicación.
- Tengan una primera aproximación al protocolo HTTP

Enunciado del proyecto:

En el mundo actual, donde la información fluye y cambia en todo momento, constantemente se están creando o modificando documentos y archivos con información que nos interesa. El estar atentos para verificar la aparición de archivos nuevos o la actualización de documentos ya existentes es cada vez más una necesidad apremiante. Por esta razón, se les ha pedido a los estudiantes del curso de redes 1, que elaboren una aplicación que permita verificar automáticamente el contenido de una serie de directorios. El programa debe monitorizar o monitorear estos directorios para notificar cuando un archivo haya sido agregado o modificado. En específico, el programa debe cumplir con los siguientes requerimientos:

R1) Identificar el nombre de los archivos nuevos, es decir, aquellos que no existían para el momento en el que se realizó la última verificación.

R2) Identificar el nombre de aquellos archivos presentes en el directorio que han sido actualizados, es decir que han sufrido alguna modificación desde la última vez que el usuario consultó la información.

Por ello se desea que usted construya un programa de nombre `verific,` que al ejecutarse esté chequeando cada X unidades de tiempo, si hay algún cambio en un conjunto de directorios en los que el usuario tiene interés, mostrando mediante mensajes de texto por pantalla, la información que se solicita en los requerimientos R1 y R2

El programa debe poder ejecutarse desde una consola de un equipo Linux, empleando la siguiente sintaxis:

```
prompt> verific [–t <num_segundos>][-d <directorio>][-a <archivo.txt>]
```

Donde:

<num_segundos>: Será un valor entero que deberá ser interpretado como el número de segundos que deben transcurrir entre dos revisiones consecutivas de los directorios monitorizados, para verificar si en ellos ha ocurrido algún cambio.

<directorio> El campo directorio corresponderá a la dirección absoluta de un directorio que se desea monitorizar, para ver si en él ha ocurrido algún cambio desde la última vez que se efectuó la revisión. Se asume que todos los directorios a trabajar en el proyecto vendrán especificados por un URL y su

contenido podrá ser accedido directamente mediante el protocolo HTTP. En otras palabras, los directorios podrán ser vistos como si fueran una página web.

<archivo.txt> será un archivo en formato de texto plano en el cual cada línea contendrá un URL que corresponderá a un directorio cuya información se desea verificar.

Ejemplo del contenido de un archivo.txt

```
http://ldc.usb.ve/~rgonzalez/ci4835/taller/proy1/dir1/
http://ldc.usb.ve/~rgonzalez/ci4835/taller/proy1/dir2/
```

Los parámetros –a –t y –d son opcionales, pero si simultáneamente se especifican los parámetros –d y –a, se verificará solamente el directorio que esta especificado en el parámetro –d. Si el parámetro –t no es especificado se puede asumir un valor por defecto de 30 segundos.

Durante la ejecución del programa `verific` constantemente los directorios configurados serán verificados y la información especificada en los requerimientos R1 y R2 deben ser mostrada mediante mensajes en texto plano, en los que se especifique qué archivos han cambiado, y cuáles archivos son nuevos. No se requiere verificar si hay algún archivo que haya sido eliminado.

Una vez que el programa se ha iniciado, podrá recibir alguno de los siguientes comandos por su entrada estándar, provenientes del teclado del equipo en el que se ejecute el programa `verific`. Cada comando estará compuesto por un carácter alfabético:

s: (salir) Para finalizar la ejecución del programa. Aquí el programa debe mostrar un mensaje de despedida por dos segundos antes de terminar

p: (pausar) Para detener temporalmente la ejecución del programa, en este caso la ejecución de la verificación sobre los directorios configurados quedará suspendida en algún punto del programa y no podrá continuar hasta que el usuario explícitamente use el comando “c”, para continua, o “s” para culminar su ejecución. Se debe indicar con un mensaje de texto en la pantalla que el programa ha sido suspendido o pausado y que se debe presionar la tecla “c” para continuar su ejecución.

c: (continuar) Para continuar la verificación de los directorios configurados, si ésta ha sido suspendida previamente.

Detalles de la implementación y otros tips

Se evaluará que el código esté bien escrito, que se sigan buenas prácticas de programación en C, que el código esté debidamente comentado y que sea legible.

La implementación del proyecto deberá escribirse en lenguaje C, compilado con gcc, y será evaluada en cualquiera de los equipos Linux del laboratorio LDC, específicamente de la sala Ernesto Leal. Si no se puede ejecutar correctamente en uno de estos equipos se considerará que el proyecto no funciona. No se corregirá en ningún otro equipo diferente.

Documentación en formato digital (Código):

Se debe entregar el código de su programa debidamente documentado, siguiendo los estándares de documentación de Doxygen, como parte de los archivos que usted coloque en aula virtual. No se

requiere que usted corra el programa Doxygen para generar la documentación, sólo que sus programas sigan los estándares de documentación especificados por Doxygen.

Su programa debe seguir las buenas prácticas de estilo de programación en C y todas las llamadas al sistema deben ser correctamente manejadas.

Condiciones de la entrega.

La entrega se podrá realizar hasta el día 20 de octubre a las 11:00 pm a través de Aula Virtual.

Los equipos deben ser exactamente de dos estudiantes, que estén ambos inscritos en el curso y en aula virtual, pueden estar en secciones de práctica diferentes.

El programa debe contener un *Makefile* que permita la correcta compilación de todos sus componentes simplemente al invocar el comando *make*. Para ejecutar los programas se debe seguir de forma estricta la sintaxis especificada en el enunciado, de no ser así se considerará que el proyecto no funciona y no será corregido.

Si usted es un integrante del grupo X (según la numeración de aula virtual) debe generar un archivo de nombre proy1grupoX.tar.gz, usando el comando tar de Linux (con las opciones cvfz), que no posea ningún directorio adicional internamente y que contenga todos los archivos que hagan falta para hacer la corrección, incluyendo el makefile. Ese debe ser el único archivo que usted coloque en aula virtual.

Información Adicional

Debido a que el proyecto debe ser realizado en equipo, se asume que cada uno de los miembros del mismo debe conocer plenamente todos y cada uno de los detalles de implementación del proyecto, ya que podrá ser interrogado al respecto durante la corrección del mismo. Aquellas personas que no muestren un dominio de los detalles del proyecto no tendrán puntos en la evaluación.

Cualquier caso de copia de proyectos será severamente castigado, no será evaluado ninguno de los proyectos involucrados y serán aplicadas las sanciones correspondientes establecidas en los reglamentos de la universidad.

GDTR1