Laboratorio 1: Sistemas Operativos

Profesor: Viktor Tapia
Ayudantes de cátedra: Muryel Constanzo y Nicolás Schiaffino
Ayudantes de Laboratorio: Ian Rossi y Luciano Yevenes

16 de agosto de 2025

1. Reglas Generales

Para la siguiente tarea se debe realizar un código programado en lenguaje C o C++. Se exigirá que los archivos se presenten de la forma más limpia y legible posible. Deberá incluir un archivo README con las instrucciones de uso y ejecución de sus programas junto a cualquier indicación que sea necesaria, y un archivo MAKE para poder ejecutar el programa.

2. Contexto

Ustedes se encuentran en pleno relajo un día, días después de haber rendido el certamen mas duro de sus vidas (aunque amigos de años superiores les aseguran que un supuesto ramo "CC" es mucho peor), cuando reciben un correo de parte de un profesor del departamento de informática solicitando su ayuda.

Cuando se contactan directamente, les explican la situación: Hace unos meses, mediante mapeo satelital avanzado, el departamento de arqueología de una universidad vecina descubrió que existe un templo sagrado en las profundidades de Siberia, Rusia. Un corto tiempo después, para quedar permanentemente en los libros de historia, el equipo que la descubrió decide ser el primero en llegar a este templo pese a no tener experiencia alguna explorando.

No habiendo pasado un día desde el comienzo de la expedición se perdió contacto con el equipo, sin embargo, unas semanas después se recibe al correo del jefe del departamento un correo sospechoso. Teniendo esperanza de que esto pueda ser un SOS el jefe del departamento le pide a su amigo, profesor de la USM, que consiga un equipo para analizar esta posibilidad, a lo cual llegamos a ustedes. Pese que le insisten al profesor que no tienen mucha experiencia en criptografía, teniendo en cuenta como les fue en el último certamen, quizá ayudarlo de igual manera no sea una mala idea.

2.1. El Archivo

El archivo recibido es un `.zip` que contiene muchas carpetas anidadas, así como archivos de texto `.txt` entre medio. A simple vista se puede apreciar que tiene la misma estructura que la instalación de un sistema operativo, pero con archivos extra.

Existen tres tipos de archivos que se pueden encontrar:

- Archivos del sistema
- Basura
- Pistas

2.2. Archivos del sistema

Estos pueden ser drivers (`.sys`), programas (`.exe`), o dynamic link-library (`.dll`).

2.2.1. Pistas

Corresponden archivos log que contienen datos útiles, como fecha y distancia a una torre de comunicación.

2.2.2. Basura

Se refiere a cualquier archivo que no caiga en alguna de las categorías anteriores.

2.2.3. Organización

Extrañamente los archivos de texto tienen nombres no descriptivos y están en carpetas desordenadas, su trabajo será determinar a cual de las 3 categorías anteriormente mencionadas pertenece, luego ordenarlos y renombrarlos según corresponda.

Para saber que a que tipo pertenece cada uno, se debe escanear el texto.

Ejemplo:

```
...
8OUdBsqMIF3izKGY24pY6vx6pCZQH3yZ
SufLO7CJ5FaO5HxJ07NF8FaZ0Bwkmu26
ciTDyYHXS2zwthphUUCxhMhRGyvhG7sy
tipo: log                          <---
1wUYScEGuAr68dgBowA1U3t8eetPCZta
FP547IQ4uHehRf27SaFvSLPLB2X8QBeI
Zvc1LhYJAyp0FDObb0dC9KwzDiiHfPQy
distancia: 395m                    <---
ls4Gij1Np3FOSFrajR2868GqZHJ8FNoh
spe63fvID6Hod58VG1ftWBlgGuYeFqzM
CFact6llACzvGlOt1RPsFzcuQl3l4zlm
...
```

Este extracto anterior correspondería a una pista.

```
...
8OUdBsqMIF3izKGY24pY6vx6pCZQH3yZ
SufLO7CJ5FaO5HxJ07NF8FaZ0Bwkmu26
ciTDyYHXS2zwthphUUCxhMhRGyvhG7sy
1wUYScEGuAr68dgBowA1U3t8eetPCZta
FP547IQ4uHehRf27SaFvSLPLB2X8QBeI
Zvc1LhYJAyp0FDObb0dC9KwzDiiHfPQy
extension: .exe                    <---
ls4Gij1Np3FOSFrajR2868GqZHJ8FNoh
spe63fvID6Hod58VG1ftWBlgGuYeFqzM
CFact6llACzvGlOt1RPsFzcuQl3l4zlm
...
```

Este extracto anterior correspondería a un programa.

2.3. Ejemplo

Se adjunta el siguiente diagrama para tener mas claridad acerca de como debiera verse el resultado final:

```
.
‘-- Output
    |-- System
    |   |-- Drivers
    |   |-- Programs
    |   ‘-- Dynamic Link-Library
    |-- Pistas
    ‘-- Basura
```

3. Tarea

Concretamente, deben escribir un programa que logre lo siguiente:

- Determinar a que categoría pertenece cada archivo
- **Mueva** cada archivo a su carpeta correspondiente en el output
- Renombre el archivo a uno descriptivo, ej: `asdolqwrk.txt` → `pista1.txt`

4. Consideraciones Específicas

- Deben hacer uso de Make, **no** de CMake.
- Se han incluido archivos de prueba para el desarrollo de su código, pero al momento de ser revisado será con archivos diferentes así como una cantidad distinta de estos.

5. README

Debe contener como mínimo:

- Nombre, RoI y Paralelo de los integrantes.
- Especificación de los algoritmos y desarrollo realizado.
- Supuestos utilizados.

6. Consideraciones Generales

- Se deberá trabajar **OBLIGATORIAMENTE** en parejas.
- Se deberá entregar a través de Github a mas tardar el día 30 de agosto a las 23:59 horas.
- Se descontarán 10 puntos por cada hora o fracción de atraso.
- Las copias serán evaluadas con nota 0 en el promedio de las tareas.
- Si se detecta uso excesivo de IA, se llamará al grupo a una interrogación presencial con los ayudantes, de la cual también será parte el profesor.

- La tarea debe ser hecha en el lenguaje C o C++. Se asume que usted sabe programar en este lenguaje, ha tenido vivencias con el, o que aprende con rapidez.
- El código debe ser entregado en forma de **1 solo archivo** `.cpp` o `.c`, nombrado en base al formato `"LAB1_ApellidoIntegrante1_ApellidoIntegrante2"`
- Los códigos serán pasados por un software anti-plagio, en caso de ser detectada copia se pondrá nota 0, hasta que se realice una reunión con los grupos involucrados.
- Pueden crear todas las funciones auxiliares que deseen, siempre y cuando estén debidamente comentadas.
- Las tareas serán ejecutadas en Linux, cualquier tarea que no se pueda ejecutar en dicho sistema operativo, partirá de nota máxima 60.
- Las preguntas deben ser hechas por Aula a través del foro o servidor de Discord. De esta forma los demás grupos pueden verse beneficiados también.
- Si no se entrega README o MAKE, o si su programa no funciona, la nota es 0 hasta la recorrección.
- Se descontarán hasta 50 puntos por:
  - Mala implementación del Makefile.
  - No respetar el formato de entrega.
  - No respetar las especificaciones del README.
  - Solicitar edición de código al momento de revisar.
  - Código poco prolijo y mal estructurado (ausencia de indentación adecuada, falta de consistencia en el estilo, nombres de variables confusos o poco descriptivos, comentarios insuficientes o irrelevantes).
- **Una vez publicadas las notas tendrán 5 días para apelar con el corrector que les revisó, después de este plazo las notas no tendrán ningún tipo de cambio**.