Laboratorio 1: Sistemas Operativos

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

19 de marzo 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. Tarea

Un profesor de la USM, reconocido por sus investigaciones en seguridad informática, estaba organizando material académico crítico para sus cursos: certámenes de múltiples años, controles históricos y tareas planificadas que debían ser utilizadas como referencia para futuras generaciones de estudiantes. Su laboratorio almacenaba estos documentos, esenciales para mantener la calidad y continuidad de la enseñanza. Sin embargo, durante una noche de trabajo intensivo, un ataque externo logró infiltrarse en los sistemas del laboratorio. Aunque ustedes actuaron rápidamente para bloquear la extracción de datos, el atacante desencadenó un protocolo de caos: los archivos fueron desestructurados, sus nombres y metadatos alterados, y algunos fragmentos corruptos con código malicioso.

2.1. Carpetas

Los archivos que tenía profesor originalmente caen en las siguientes categorías:

- Certámenes
- Controles
- Tareas

2.2. Semestres

Dentro de cada una de estas, existían carpetas que separaban los archivos según su semestre correspondiente: 2024-2, 2024-1, 2023-2, etc.

2.3. Por cada categoría

2.3.1. Certámenes

Estos pueden corresponder al primer segundo y tercer certamen: C1, C2 y C3.

2.3.2. Controles

Al igual que con los certámenes estos pueden corresponder al primer segundo y tercero: Q1, Q2 y Q3.

2.3.3. Tareas

Estas pueden ser :

- Onda Certera
- Láser vs. Plasma
- Quantumanía
- ÁtomoX
- BitFase
- Red-Arácnida
- Debug & Dragons
- Thread Wars
- Crypto Knight
- Bugzilla
- Stack Attack
- Compile & Conquer
- AlgoRhythm
- Cyberkarma

2.4. Organización

Durante el ataque, a los archivos se les ha borrado el nombre, dejándolos con uno no descriptivo.

Para ayudar al profesor, deben mover los archivos de la carpeta en la cual están desordenados, a la que corresponda según lo establecido en los puntos anteriores, renombrándolos además según corresponda.

Pese a que estos no son legibles directamente en formato `.txt`, al abrirlos como si lo fuesen se puede obtener la información necesaria para poder organizarlos al buscar.

Ejemplo:

```
...
8OUdBsqMIF3izKGY24pY6vx6pCZQH3yZ
SufLO7CJ5FaO5HxJ07NF8FaZ0Bwkmu26
ciTDyYHXS2zwthphUUCxhMhRGyvhG7sy
tipo: certamen                    <---
1wUYScEGuAr68dgBowA1U3t8eetPCZta
FP547IQ4uHehRf27SaFvSLPLB2X8QBeI
XFZyAJx2x4rx3twlcsYUGiS5QWbWaH6L
AsAE8cFffbdhxkodpSex4TSUYcbm2zfR
numero: 2                         <---
wSTwFE6rhY4uKggELUbbr52AyAZKUmBq
WHCAG3fWlXAAhupP32VQLxeLBiNBtzpH
Zvc1LhYJAyp0FDObb0dC9KwzDiiHfPQy
semestre_publicacion: 2023-1      <---
ls4Gij1Np3FOSFrajR2868GqZHJ8FNoh
spe63fvID6Hod58VG1ftWBlgGuYeFqzM
CFact6llACzvGlOt1RPsFzcuQl3l4zlm
...
```

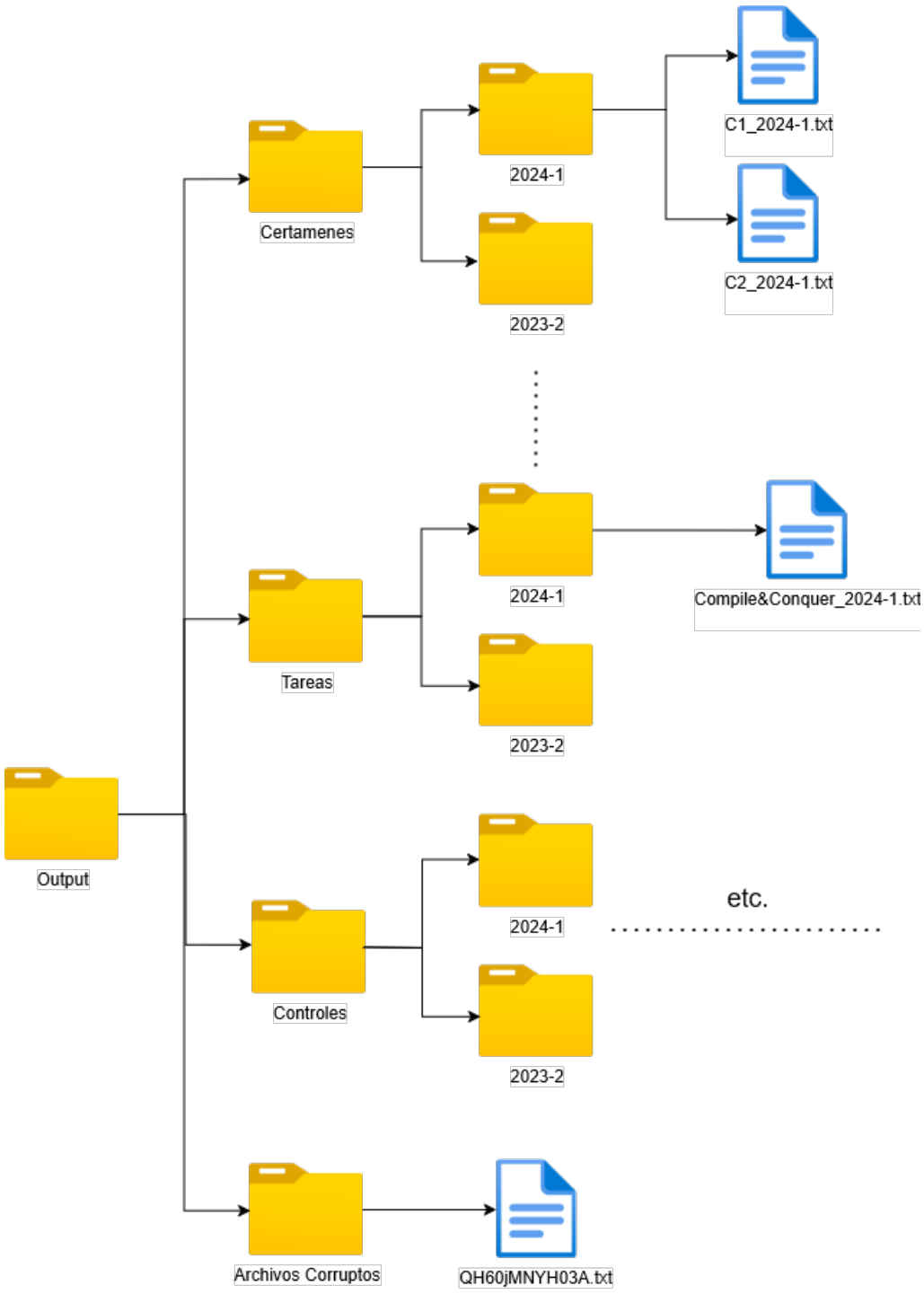
Este extracto anterior correspondería al Certamen 2 2023-1

2.5. Datos Corruptos

Finalmente, si es que algún archivo tuviera un tipo, número o semestre inválido, se debe considerar como corrupto, moviéndose en cambio a una carpeta designada especialmente para este tipo de archivos.

2.6. Ejemplo

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



3. Consideraciones Específicas

- No necesariamente van a estar presentes todas las evaluaciones y tareas para cada semestre, por ejemplo, puede que para el 2024-1 encuentren el C1 y C3, pero no el C2.
- 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.

4. BONUS (20 pts extras)

Al inspeccionar más a fondo los directorios recuperados tras el ataque, se percataron que existían algunos archivos que no podían leerse a simple vista. Al abrirlos, parecían contener únicamente cadenas extrañas e ilegibles. Tras una breve investigación, descubrieron que varios de esos archivos se encontraban codificados en Base64. Por ello, como parte **OPCIONAL** de esta tarea, se solicita implementar un mecanismo que identifique y decodifique estos archivos Base64, restituyendo así su contenido original y permitiendo al profesor recuperar el material completo. Esto incluye:

- Detección de archivos que contengan contenido en Base64.
- Decodificación del contenido usando creando una función en C/C++.
- Verificación posterior de que el contenido decodificado corresponda a información válida y no a datos corruptos.

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 Aula a mas tardar el día Domingo 6 de Abril a las 23:59 horas.
- Se descontarán 10 puntos por cada hora o fracción de atraso.
- La nota máxima obtenible, si se realiza también el bonus, es de 120.
- Las copias serán evaluadas con nota 0 en el promedio de las tareas.
- 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 50 puntos por:
    - Mala implementación del Makefile.
    - No respetar el formato de entrega.
    - 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**.