202 – Why is not it playing?

Team Information

Team Name : kimbabasaksaksak

Team Member : Jaeheon Kim, Donghyun Kim, Soyoung Yoo, Minhee Lee

Email Address : uaaoong@gmail.com

Instructions

Description The memory card installed in the video recording device is damaged, so only some of the data in the memory can be obtained. The encoding information and video structure of file are found in the remaining data, but the image is not confirmed. The video recording device consists of a file system including a security element that can play and back up videos only through the manufacturer's exclusive viewer. The video, voice, and time information are recorded in this file, and one book is recorded in the video file.

Target	Hash (MD5)
final-final-problem.img	317492066299DE92BA2A2D718785160A

Questions

1) Submit the title and edition information of the book recorded in the video. (200 points)

Teams must:

- Develop and document the step-by-step approach used to solve this problem to allow another examiner to replicate team actions and results.
- Specify all tools used in deriving the conclusion(s).

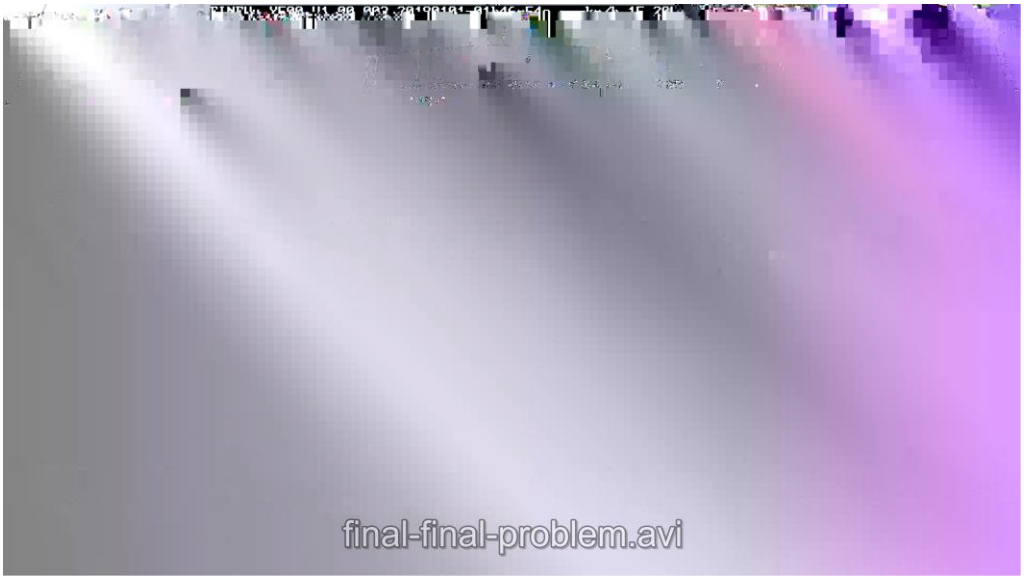
Tools used:

Name:	010 Editor	Publisher:	SweetScape
Version:	13.0.2		
URL:	https://www.sweetscape.com		

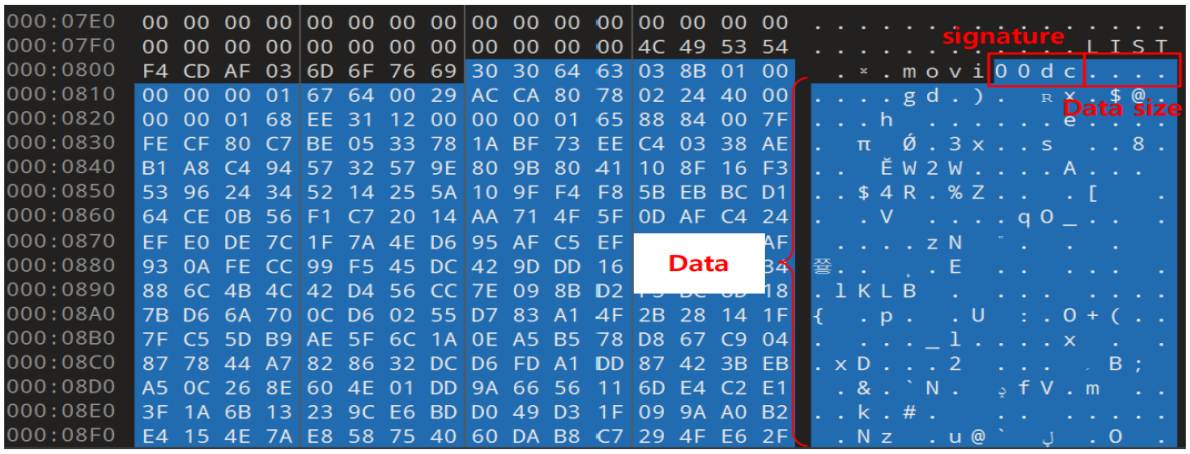
Step-by-step methodology:

Q1. Submit the title and edition information of the book recorded in the video. (200 points)

주어진 파일을 재생하면 아래와 같이 정상적으로 재생되지 않는 모습을 볼 수 있다.



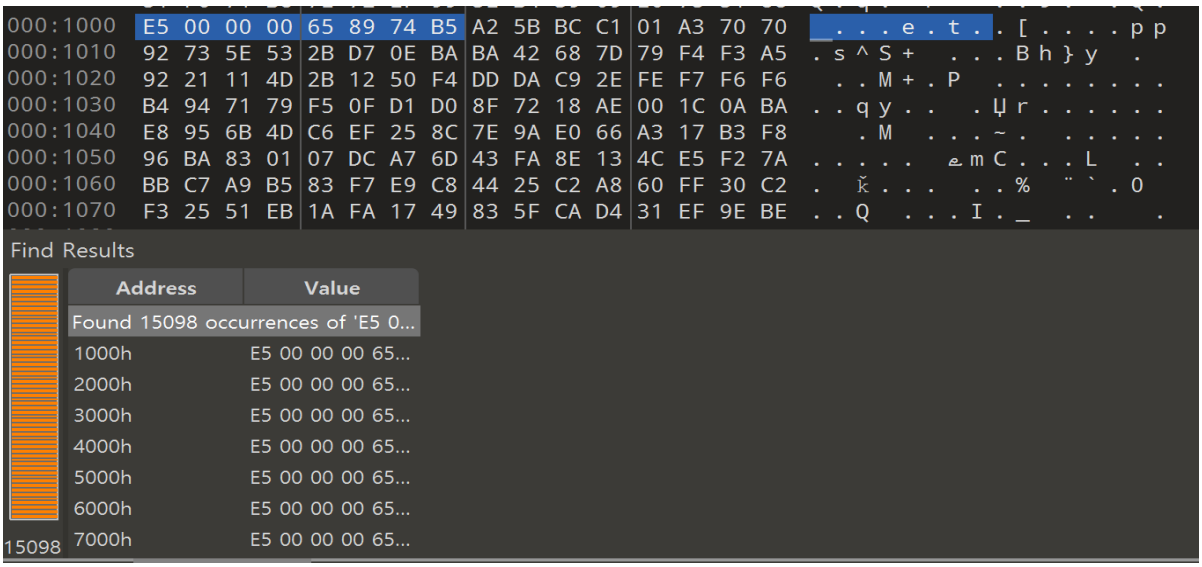
final-final-problem.img 파일을 확인해보면 RIFF AVI 시그니처를 통해 해당 파일이 AVI 파일임을 알 수 있다. AVI 파일에서 스트림의 실제 데이터가 존재하는 부분은 movi 리스트이다. movi 리스트에서 파일 내에 존재하는 모든 구조체의 시그니처를 확인한 결과, 00dc, 01wb, 02tx로 총 3개의 시그니처가 존재한다. 각 시그니처는 video frame, audio frame, subtitle frame을 의미한다.



비디오 프레임 구조체를 확인하면 데이터 크기 값과 실제 비디오 프레임 데이터의 크기가 일치하지 않는 것을 확인할 수 있다. 아래는 movi 리스트의 첫 3개 비디오 프레임의 크기 값과 실제 크기를 작성한 것이다.

	video frame size	실제 video frame size
0	101123 byte	101324 byte
1	28519 byte	28576 byte
2	33021 byte	33086 byte

비디오 프레임 크기가 커질수록 데이터 크기로 명시된 값과 실제 비디오 크기와의 차이가 더 커진다. 이를 통해 일정한 길이 단위마다 삽입되는 dummy data가 있을 것으로 추정하였다. final-final-problem.img 파일의 0x1000 byte마다 `'E5 00 00 00 65 89 74 B5'` 가 삽입되어 있는 것을 확인할 수 있었다.



movi 리스트에서 video frame 만을 추출하되 'E5 00 00 00 65 89 74 B5' 값을 공백으로 치환하는 코드를 이용해 원본 비디오를 복원하였다.

```
file = "final-final-problem.img"

with open(file, "rb") as f:
    data = f.read()
    data = data.replace(bytes.fromhex('E5 00 00 00 65 89 74 B5'), b'')
data_len = len(data)
video = open("test.h264", "wb")


def extract_video():
    idx = 0
    while True:
        video_idx = data.find(bytes.fromhex("30 30 64 63"), idx)
        size = int.from_bytes(data[video_idx + 4 : video_idx + 8], "little")
        if video_idx == -1:
            break
        else:
            video.write(data[video_idx + 8 : video_idx + 8 + size])
            idx = video_idx + 4


extract_video()
```

*** Q1 Answer: Introduction to Spectroscopy Fourth Edition**