───────────────────────────────────────────────

**📄 CTF WRITE-UP REPORT**

**🔐 SECURITY FOOTAGE – TRYHACKME ROOM**

───────────────────────────────────────────────

👤 Name: Unaiza Fatima

🎓 Program: BS Cyber Security – 4th Semester

🏫 Institute: Air University

🧠 Platform: TryHackMe (www.tryhackme.com)

📅 Date Solved: 6/27/2025

🌐 Room Link[: https://tryhackme.com/room/securityfootage](:%20%20%20https:/tryhackme.com/room/securityfootage)

📁 Report Type: Digital Forensics / Packet Capture Analysis

📌 Tools Used: Wireshark, binwalk, ffmpeg, xdg-open

**🕵️‍♀️ Security Footage CTF – My Step-by-Step Walkthrough**

So here’s how I tackled the CTF challenge involving a .pcap file that **seemed to contain fragments of a security footage** not an actual video, but a bunch of images. Later, I converted those into a playable video which eventually revealed the flag. I’ve broken down what I did in simple clear steps even if you’re new you can follow along.

**Tools I Used (In Simple Terms)**

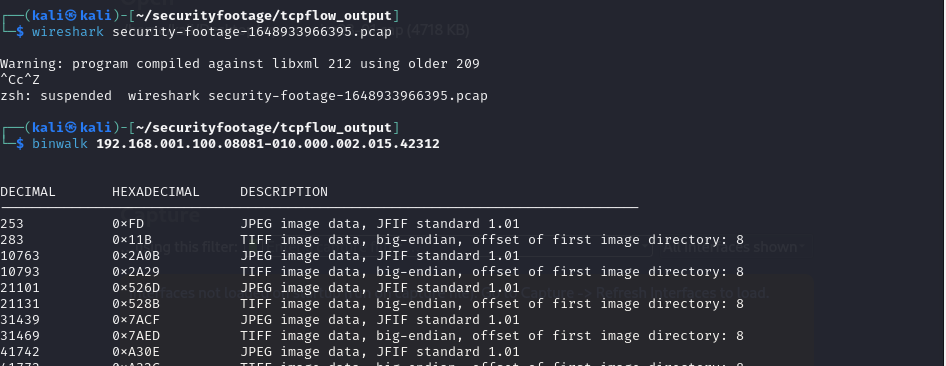
* **Wireshark** – to open and explore the .pcap file (basically a recorded network traffic file).
* **binwalk** – to scan the file and check if it had hidden images or media inside.
* **ffmpeg** – to fix and rebuild the broken video from all the images.
* **xdg-open** – a Linux command to quickly open media files without hunting through folders.

**✅ What I Did – Step-by-Step**

**🔍 Step 1: Opened the PCAP File in Wireshark**

wireshark security-footage-1648933966395.pcap

I launched Wireshark to check what kind of data was flowing in the network capture. I focused on HTTP and TCP streams. It became obvious that this was a **video stream sent frame by frame** basically lots of JPEG images.



**📦 Step 2: Scanned for Hidden Files with binwalk**

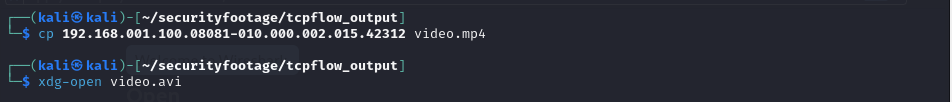
binwalk 192.168.001.100.08081-010.000.002.015.42312

I ran binwalk on one of the extracted files, and boom it showed **JPEG file headers** inside. This confirmed that actual image data was embedded, just not properly formatted.

**📂 Step 3: Renamed the File to MP4**

cp 192.168.001.100.08081-010.000.002.015.42312 video.mp4

Since the file seemed to be media, I just **renamed it with a .mp4 extension** to trick the system into treating it like a video. Sometimes that's all it takes!



**🎬 Step 4: Tried Opening It with xdg-open**

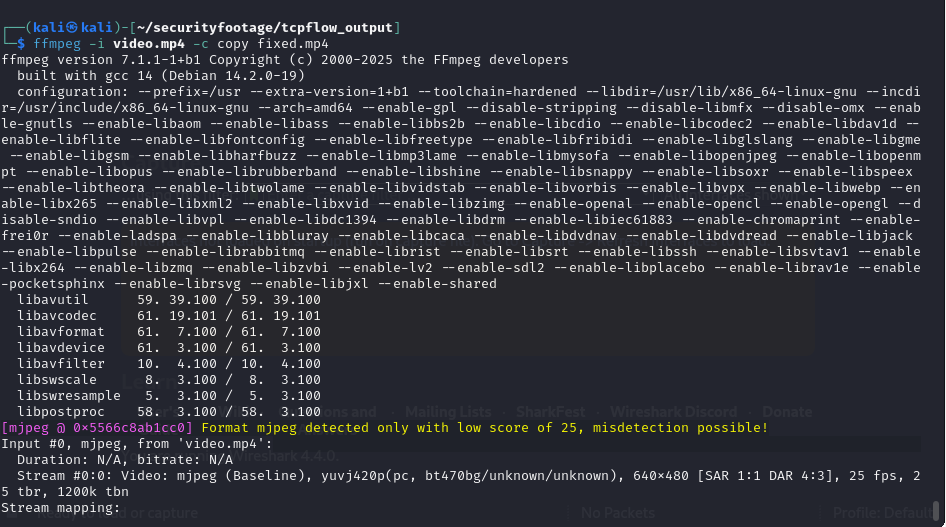
xdg-open video.avi

I also tested it with .avi format just to be sure. Didn’t work. But this command helped me test media files quickly in Linux, no extra clicking needed.

**🛠 Step 5: Fixed the Video with ffmpeg**

ffmpeg -i video.mp4 -c copy fixed.mp4

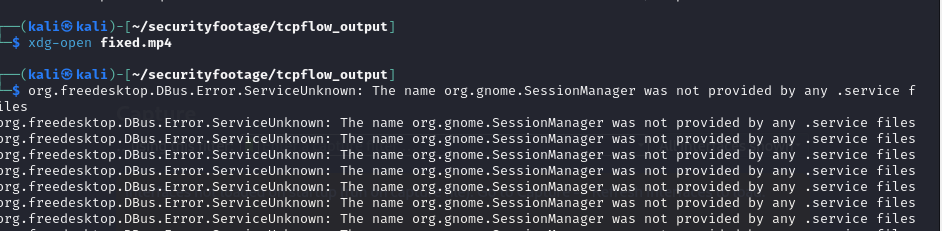
The renamed video file was broken, so I used **ffmpeg** to create a fixed copy. It copied the video data without changing it and fixed the formatting. This was key to making the file playable.



**▶️ Step 6: Played the Fixed Video**

xdg-open fixed.mp4

Finally, I opened the fixed video and there it was! The actual **security footage** played like a short clip, and at one point, the **flag appeared** on a whiteboard or piece of paper in the frame.





**💡 What I Learned**

* Wireshark is powerful always check for streams in .pcap files.
* binwalk can reveal hidden or embedded files you’d never spot otherwise.
* ffmpeg is the GOAT when it comes to fixing media files.
* Don’t give up if a file doesn’t open **try changing the extension first**!