

# **SIT282/SIT703 Computer Forensics and Investigations**

## **Workshop Session 5**

In the previous session, you have learned how to acquire and analyze a forensic image using tools.

Through this session, you will learn how to recover data from a binary file with a complicated structure. You will practice two forensic software tools – file and HxD Hex Editor. By solving forensic tasks, you will be able to identify the real content of a file, restore the appropriate file header, and manipulate data blocks manually. Ubuntu VM is used in this session.

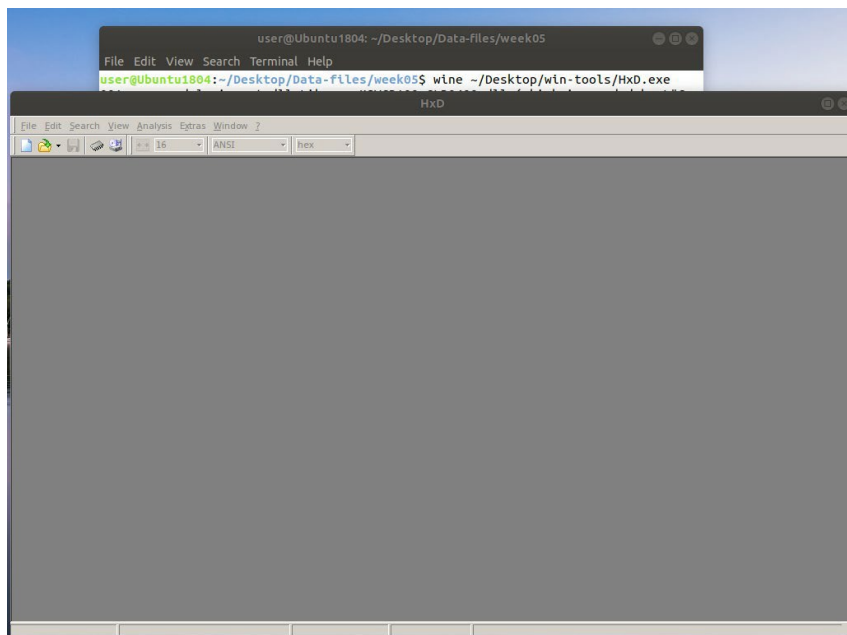
### **Learning Objectives**

1. Demonstrate that you can identify various pieces of information displayed in the panes of a hexadecimal editor.
2. Identify the relationship between the file representation in hexadecimal and the ASCII code.
3. Demonstrate you can use both the hex and string editing panel in a hexadecimal editor to create, modify and display the contents of a file.
4. Explain the meaning of a ‘binary’ file and how it is represented in a hexadecimal editor.
5. Practice using a tool that can be used to distinguish between different file types.
6. Define the term “magic number” and how it can be searched for within a file.
7. Demonstrate that you can repair broken file headers.

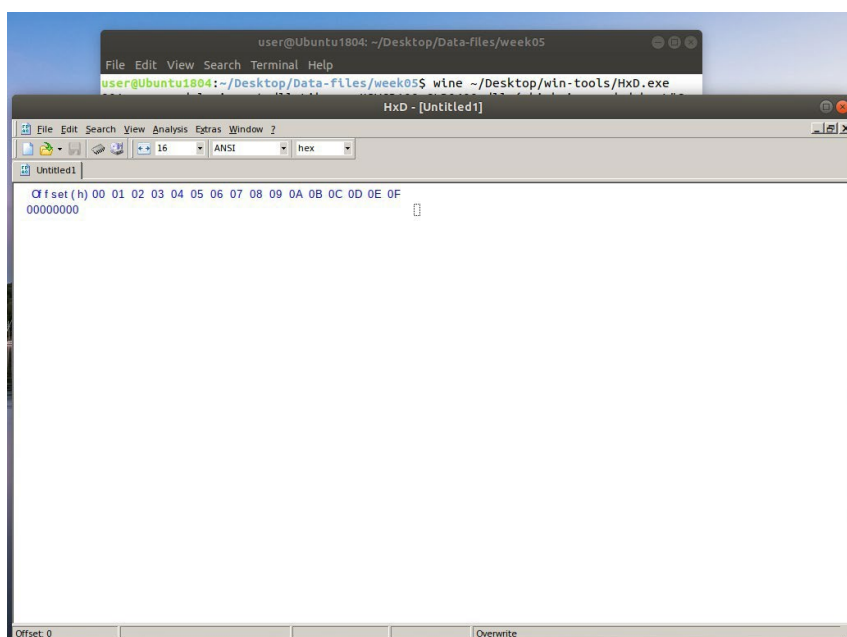
### **1. Manipulating Data by Using HxD**

Hex editors such as HxD are commonly used to manually edit data. In fact, Hex editors are used in the same manner as any other editor programs – notepad, Word and so on. The most commonly used features are “make a new file”, “insert”, “overwrite”, “copy” and “paste”.

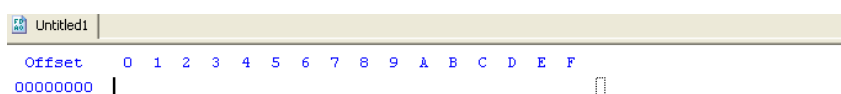
To practice these functions, we use HxD editor which has been installed in the virtual machine. Because it is a Windows executable, we need to use the emulator “wine” to run the editor. First, open a “Terminal” and change the directory to “Desktop/Data-files/week05/”. Launch the HxD editor by typing the command “wine ~/Desktop/win-tools/HxD.exe”, you will see the following screenshot:



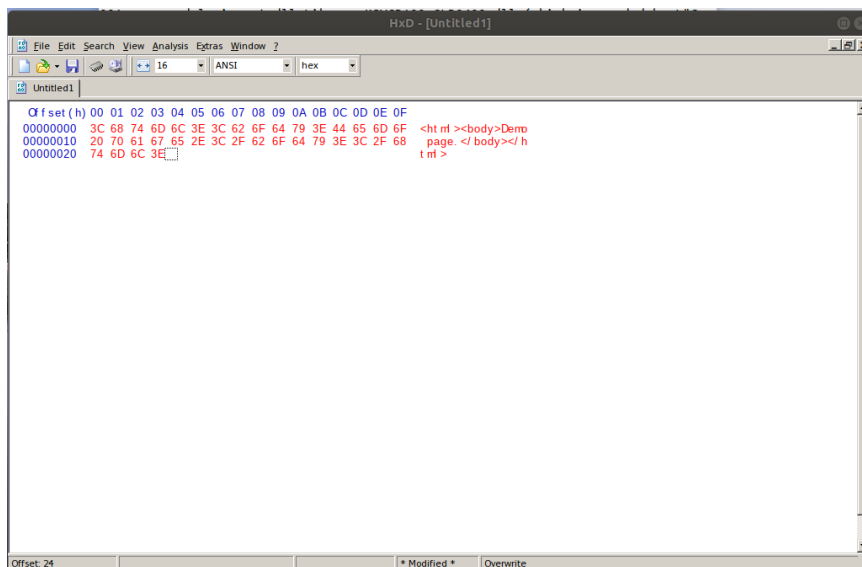
Now open an empty document by pressing “Ctrl-N” or File>New. You will see



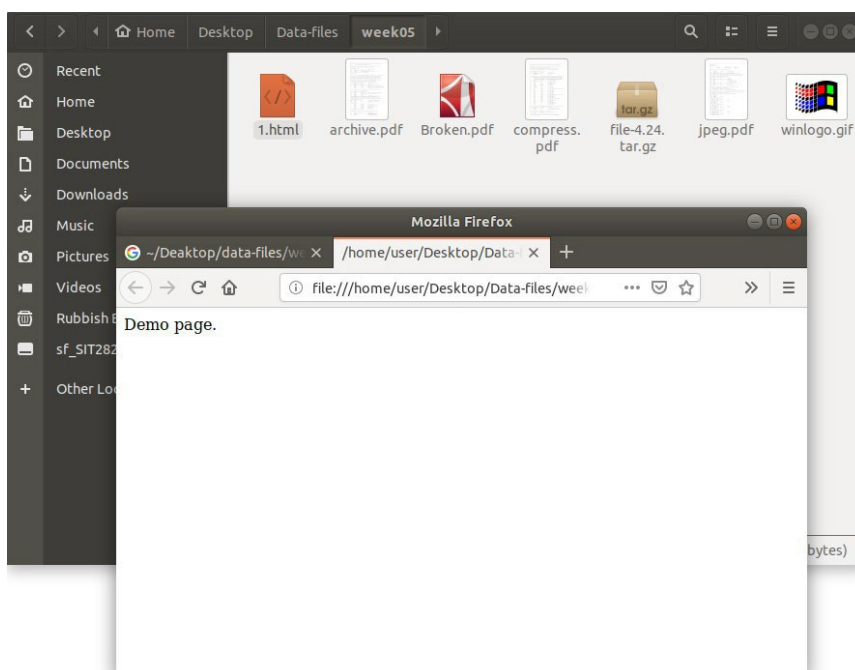
The panel under number 0 to F (or 00 to 0F) is called “hex editing panel” and the panel to its right is “string editing panel”. You will understand the difference between them by completing this section. The editing mode can be either “Insert” or “Overwrite”. The current editing mode is displayed by an indicator (at the bottom). The editing mode is “Overwrite” in the above screen. To toggle to the other editing mode, press the “Insert” key on the keyboard and you will see a slightly different cursor blinking:



Type in the string – “<html><body>Demo page.</body></html>” in the editor as this screen indicates:

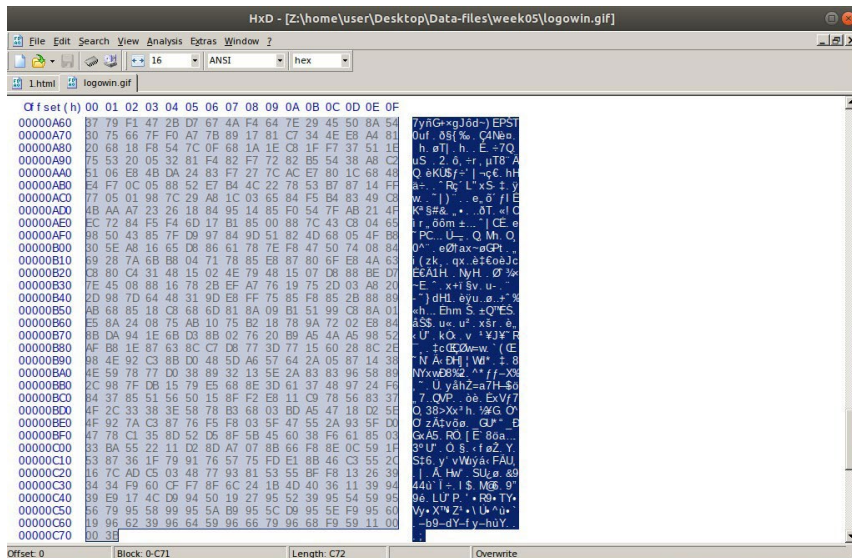


Save this file as a HTML file named “1.html” to the folder “Desktop/Data-files/week05” in the virtual machine. Open the file in Mozilla Firefox browser.

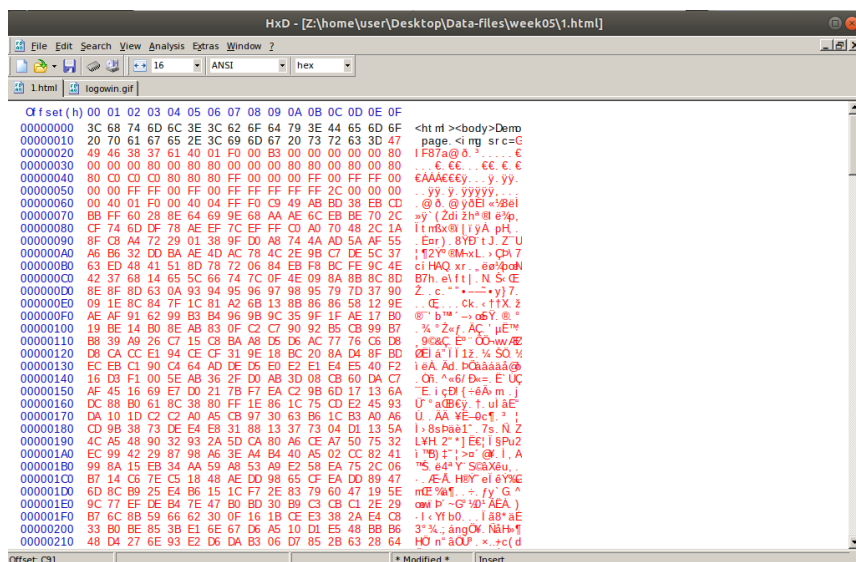


Locate the image file “logowin.gif” in the “~/Desktop/Data-files/week05” folder and include this picture in the “1.html” file by adding the string ‘<img src=“logowin.gif”>’ as shown in the following screen. Save the file and refresh the page in the browser.

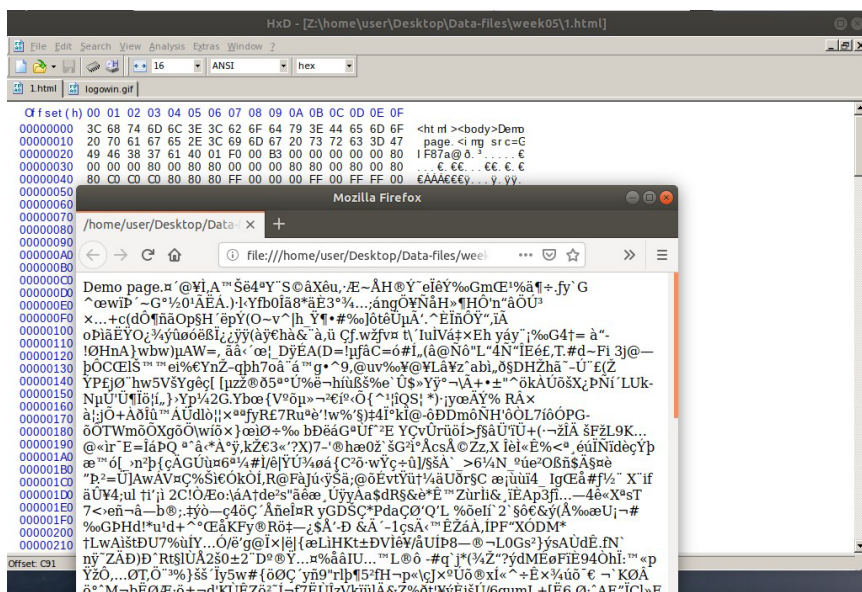




Switch to the file “1.html”. Replace the file name “logowin.gif” by the actual content of the image file, i.e., delete the string “logowin.gif” and press “Ctrl-V” in the hex-editing panel.



Save the HTML file and refresh the page in the browser. You should see many ASCII characters.



Now, switch to the “1.html” file in the HxD editor. Select the content you just pasted, cut the highlighted content, and save it in a separate file. Your newly saved file (e.g. logowin2.gif) should be the same as the original file “logowin.gif”. Use the program md5sum to validate your result.

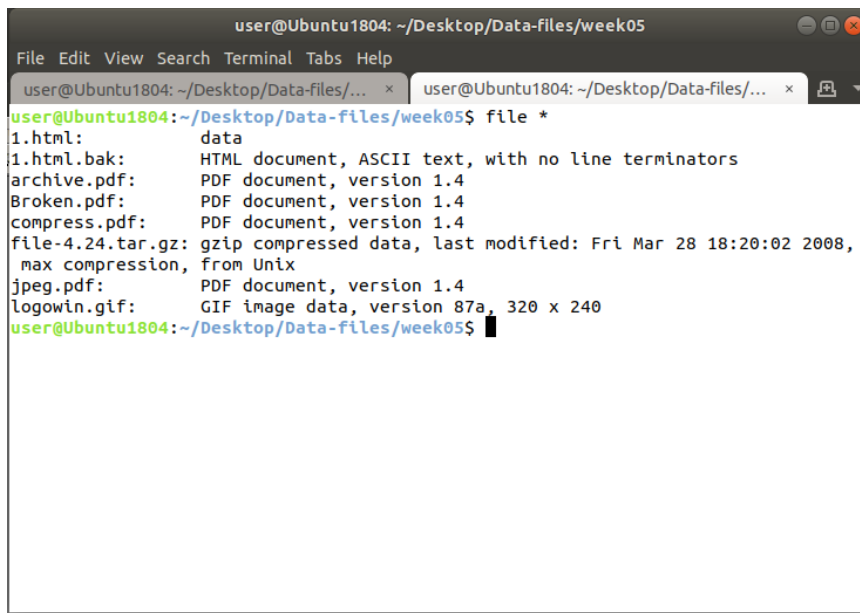
## 2. Introducing the Linux tool file

The Linux tool file is a program used to automatically distinguish file types. It was developed on Unix systems in the last century when a user could create some files with arbitrary extensions. If the user wanted to use a file but forgot the file type, they had to try all possible programs to open this file. The file program was designed to tell the user which program should/could be used to open the file.

Nowadays, many operating systems by default don’t show common extensions to users to prevent alteration. This fool-proof mechanism is very primitive, and digital forensic investigators shall never be fooled by a modified file header.

We will first look at what the file program actually does. Type the command “**file** \*\*” and you will see the following screen:





```
user@Ubuntu1804: ~/Desktop/Data-files/week05
File Edit View Search Terminal Tabs Help
user@Ubuntu1804: ~/Desktop/Data-files/... x user@Ubuntu1804: ~/Desktop/Data-files/... x
user@Ubuntu1804:~/Desktop/Data-files/week05$ file *
1.html:      data
1.html.bak:  HTML document, ASCII text, with no line terminators
archive.pdf: PDF document, version 1.4
Broken.pdf:  PDF document, version 1.4
compress.pdf: PDF document, version 1.4
file-4.24.tar.gz: gzip compressed data, last modified: Fri Mar 28 18:20:02 2008,
               max compression, from Unix
jpeg.pdf:    PDF document, version 1.4
logowin.gif: GIF image data, version 87a, 320 x 240
user@Ubuntu1804:~/Desktop/Data-files/week05$
```

4 PDF files, 1 GIF image, 1 html with data, 1html backup file, and 1 gzip archive file are reported. Now we focus on the file “file-4.24.tar.gz”, as its content is reported as a gzip compressed data.

The mechanism of the file program is extremely simple – it searches “magic numbers” in binary form through an input file. The definition of “magic numbers” is straightforward. Open file “compress.pdf”, and you will see the following text from the second paragraph on the first page.

```
# gzip (GNU zip, not to be confused with Info-ZIP or PKWARE zip archiver)
#   Edited by Chris Chittleborough <cchittleborough@yahoo.com.au>, March 2002
#
#   * Original filename is only at offset 10 if "extra field" absent
#   * Produce shorter output - notably, only report compression methods
#     other than 8 ("deflate", the only method defined in RFC 1952).
0    string      \037\213      gzip compressed data
```

The “magic number” for gzip files is the string of an Oct value “\037\213” from the offset “0” byte. That is, the number “0” indicates the offset; the value “string” indicates this magic number can be searched as a string; “\037\213” is the magic number but in Oct value and the string “gzip compressed data” corresponds the first two bytes which you saw in the above screenshots. In order to observe these two bytes in a hex editor, we convert these two Oct numbers to the Hex value (“0x1F8B”) which you can find as the first two bytes of the gzip file. (See Screenshot below)

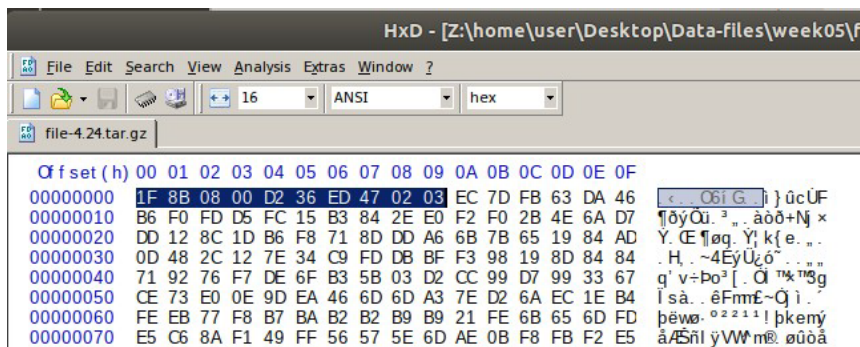
Spend 5 minutes to read through the document “compress.pdf” to find where the following strings are from

- “from Unix”
- “max compression”

The document reads

```
>8    byte    2      \b, max compression
>9    byte    =0x03  \b, from Unix
```

To really understand what they actually mean, we use HxD Hex Editor to open the file “file-4.24.tar.gz”. It looks like:



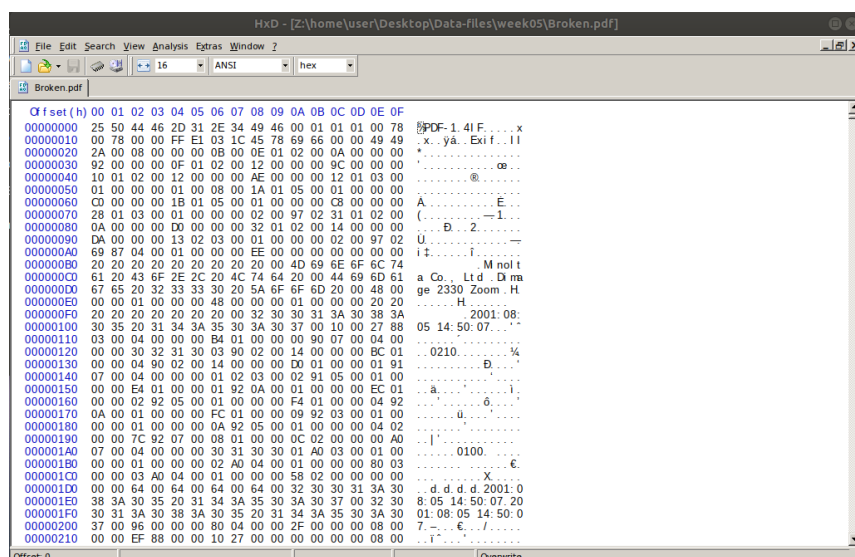
Pay extra attention to the highlighted part. The first two bytes of the Hex value “0x1F8B” is equivalent to “\037\213” – you may use the scientific calculator “calc” to verify this; the eighth offset is of the value “0x02”, which means “max compression”; and the ninth offset is of the value “0x03”, which means “from Unix”.

Now, unzip the file “file-4.24.tar.gz” using “gzip -d -k file-4.24.tar.gz”. You get the file-4.24.tar file. Find out the “magic number” for this tar file. The document you need to read is “archive.pdf”. (Hint: the magic number is not in the very beginning of the tar file. Search “archive format” in “archive.pdf”. Then open .tar in HxD, hexadecimal values “75 73 74 61 72” or “ustar” in ASCII.)

### 3. Repairing Broken File Headers

Occasionally, file headers can provide very misleading information. For instance, many picture files are often given some other type headers. In this workshop, we focus on JPEG files. Many JPEG files start with a Hex string “0xFFD8”.

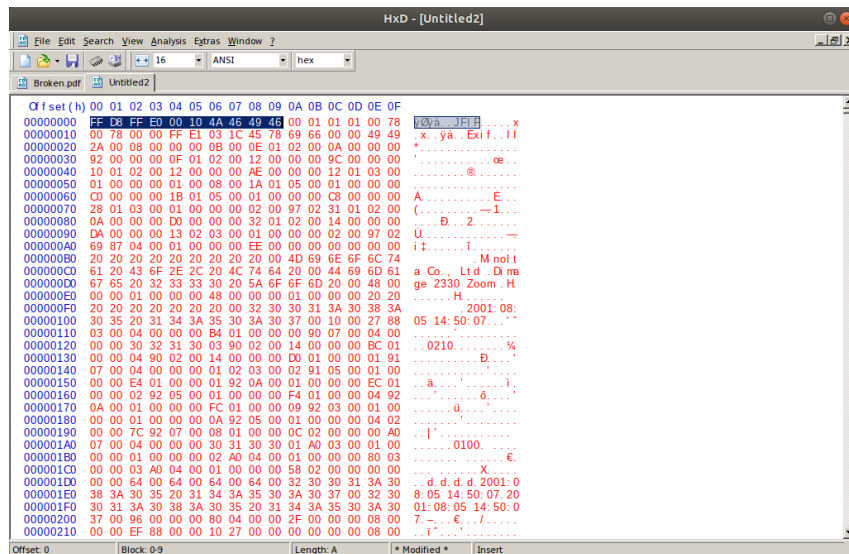
Focus on the file “Broken.pdf”. The file program reported it as a PDF file based on a magic number. However, this file is not a legitimate PDF file. You can verify it by trying to open with any PDF reader programs. So, we take a closer look at the file. Open “Broken.pdf” in HxD, it shows:





Focus on the first two lines. The first 8 bytes (from offset 0 to offset 7) indicate that the file is a PDF file. However, the second line has an “Exif” string which indicates that it is an Exif header embedded in a JPEG file. Therefore, we need to restore the right header. Make a copy of this file so that we can modify it.

According to the textbook, a JPEG header is normally of the hex value “0xFFD8FF E000104A464946”. So we replace the first 10 bytes of the replica of the file “Broken.pdf” by using the right Hex value. Note: Your HxD should be in the “Overwrite” mode when you try to replace contents.



Save it in HxD and rename the file to “Broken.jpg”. Then you should see the following picture.

