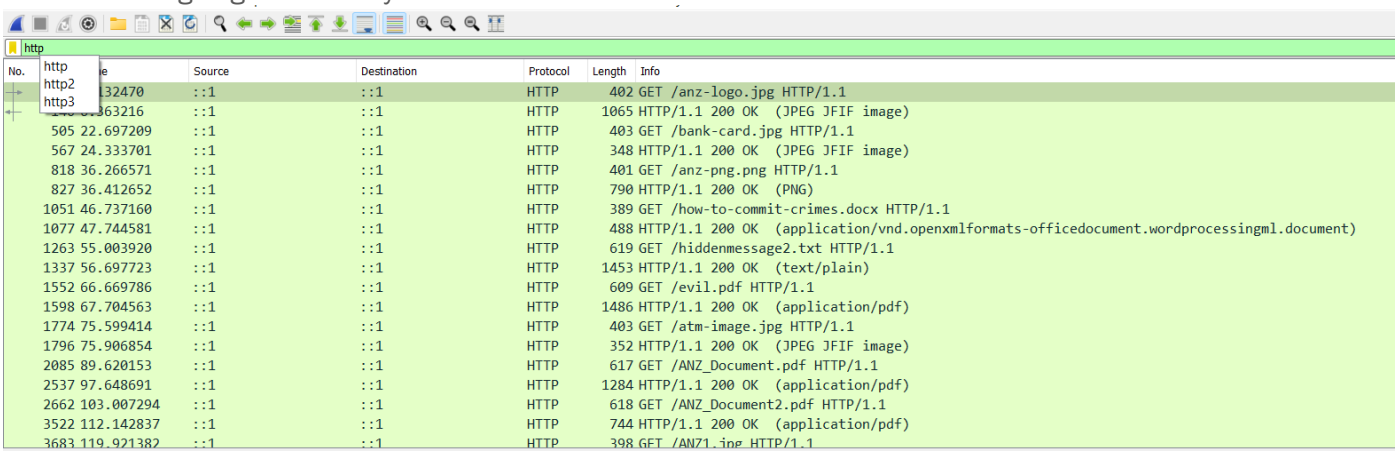
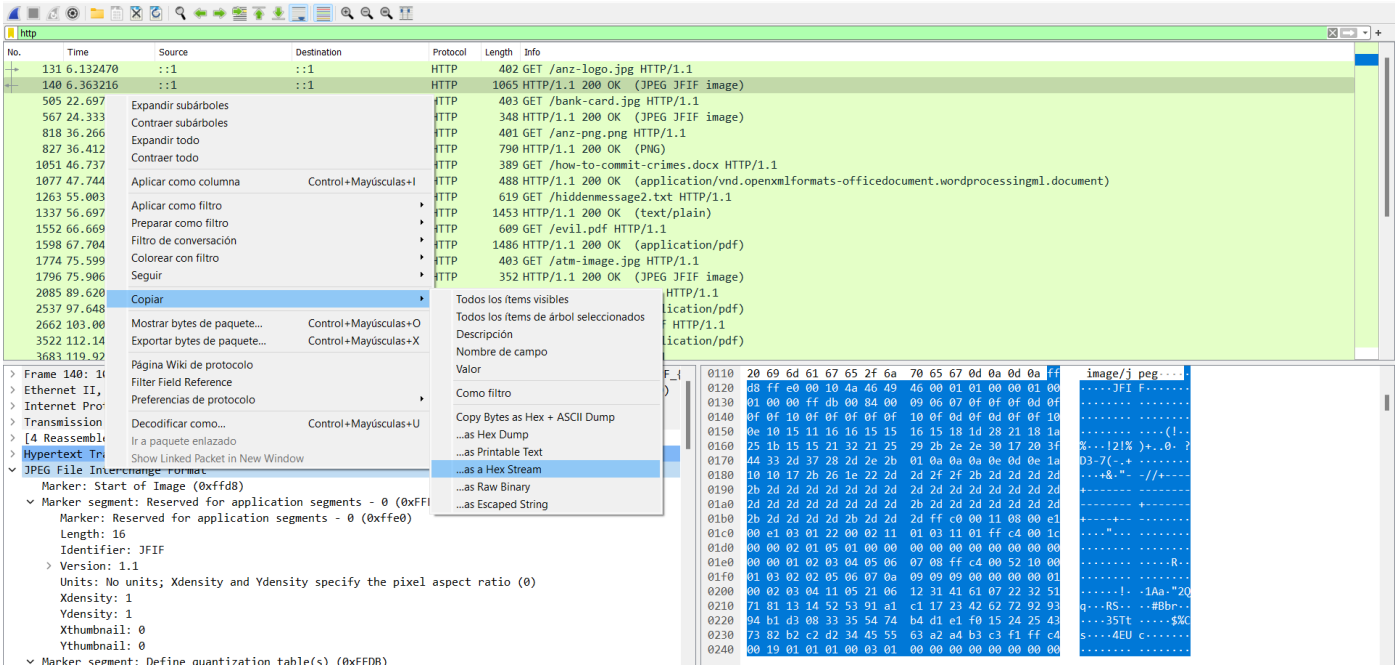
Sub-task 1:

- *anz-logo.jpg and bank-card.jpg are two images that show up in the users network traffic.*
- *Extract these images from the pcap file and attach them to your report.*

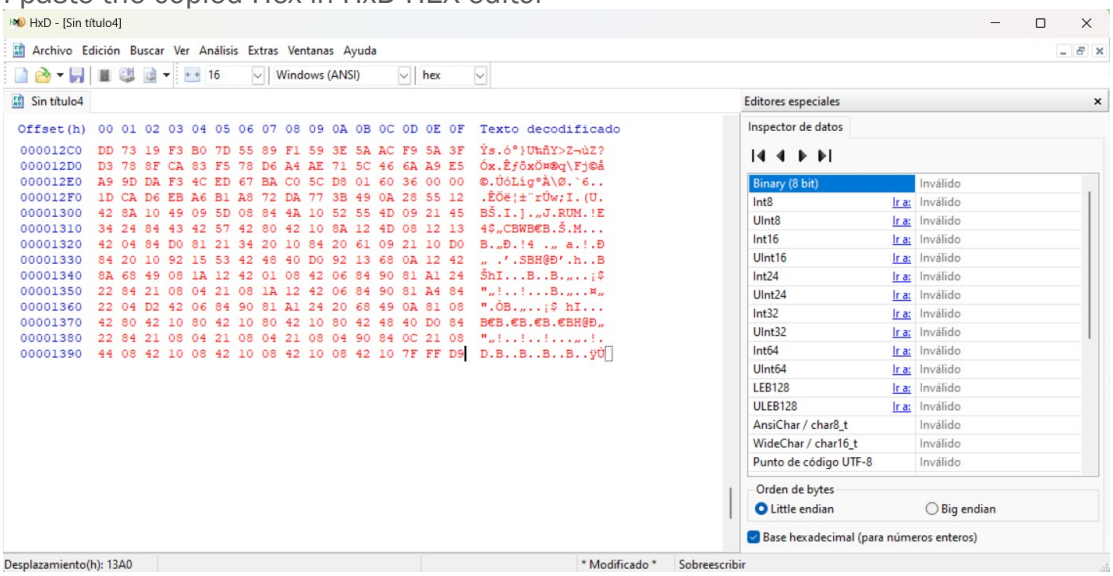
First of all I'm going to filter only the HTTP traffic:



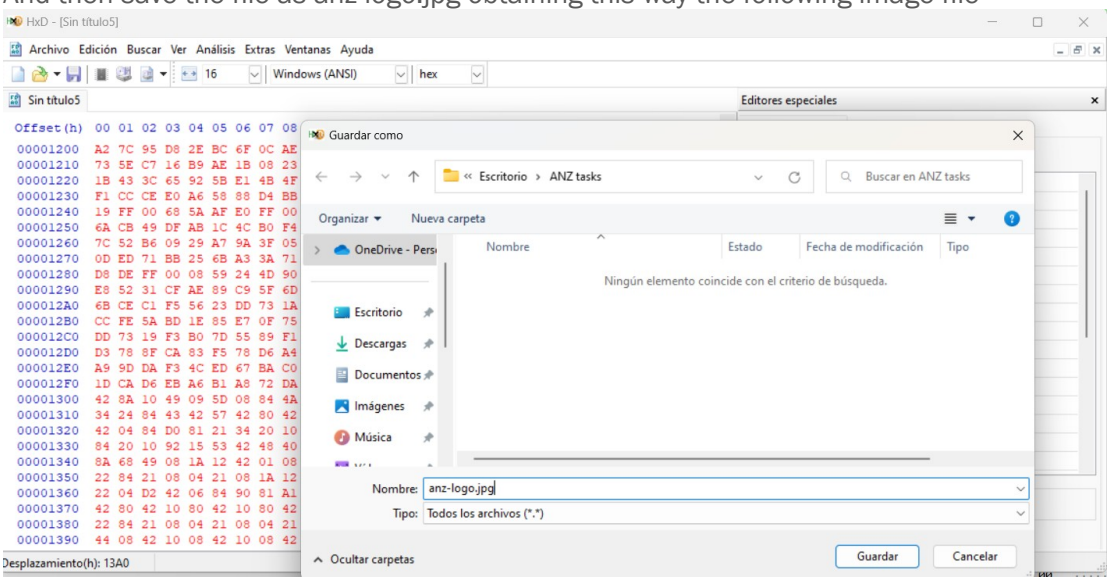
For the anz-logo.jpg look after the response of that specific request (line #140) and copy its code as a HEX stream



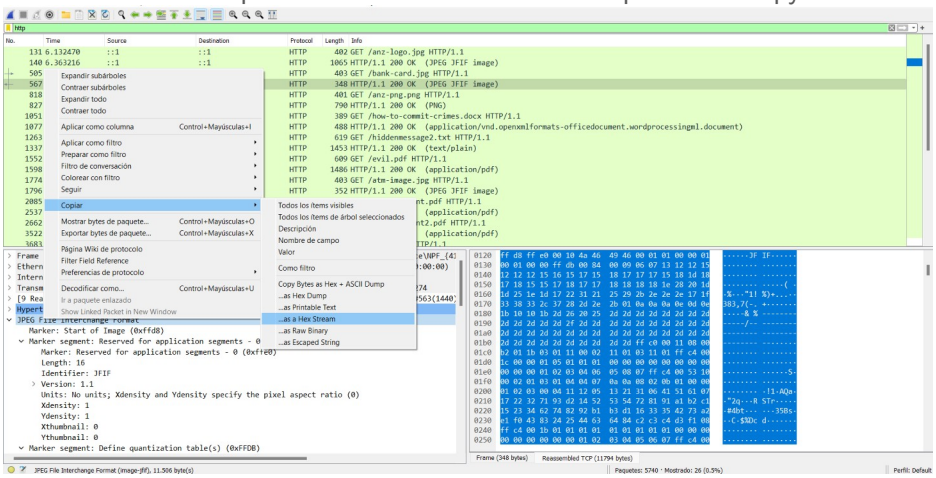
I paste the copied Hex in HxD HEX editor



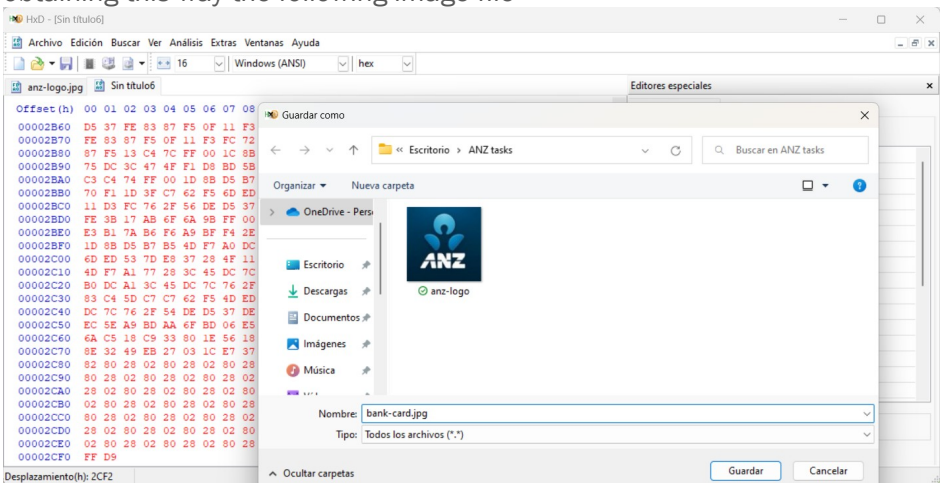
And then save the file as anz-logo.jpg obtaining this way the following image file



For the bank-card.jpg image the same steps must be followed
First search the response to the bank-card GET request and copy the file code as a Hex Stream



Paste the copied Hex in HxD HEX editor and then save the file as anz-logo.jpg obtaining this way the following image file

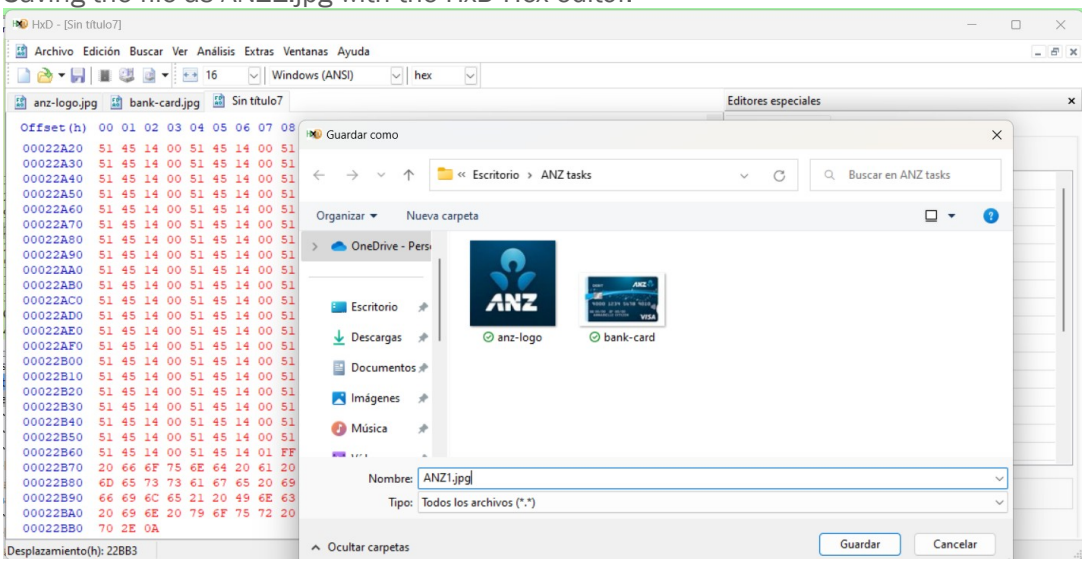


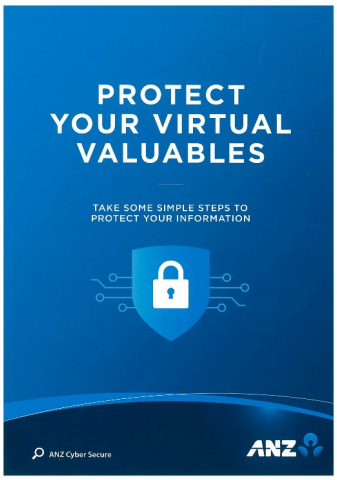
Sub-task 2:

- *The network traffic for the images "ANZ1.jpg" and "ANZ2.jpg" is more than it appears.*
- *Extract the images, include them and mention what is different about them in your report.*

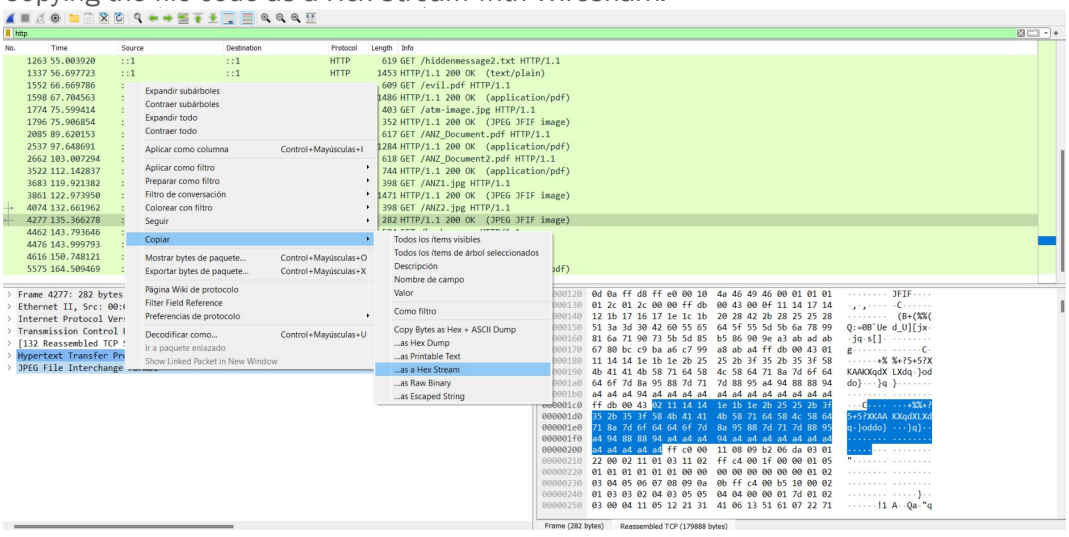
Following the same steps explained in the Sub-task 1, copy the files as a Hex Stream and save the files in the HxD Hex editor and obtain the images

Saving the file as ANZ1.jpg with the HxD Hex editor.





Copying the file code as a Hex Stream with Wireshark.



Differences between files:

We can notice a few differences between files such as the file size expressed at the “File Data” field among others.

```
Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Fri, 16 Aug 2019 00:49:42 GMT\r\n
    Server: Apache/2.4.6 (CentOS)\r\n
    Last-Modified: Thu, 15 Aug 2019 14:16:05 GMT\r\n
    ETag: "2bd8e-5902883a60136"\r\n
    Accept-Ranges: bytes\r\n
  > Content-Length: 179598\r\n
    Keep-Alive: timeout=5, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: image/jpeg\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 2.704316000 seconds]
    [Request in frame: 4074]
    [Request URI: http://localhost:8000/ANZ2.jpg]
    File Data: 179598 bytes
```

```
Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Fri, 16 Aug 2019 00:49:30 GMT\r\n
    Server: Apache/2.4.6 (CentOS)\r\n
    Last-Modified: Thu, 15 Aug 2019 14:14:04 GMT\r\n
    ETag: "22bb3-590287c71be98"\r\n
    Accept-Ranges: bytes\r\n
  > Content-Length: 142259\r\n
    Keep-Alive: timeout=5, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: image/jpeg\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 3.052568000 seconds]
    [Request in frame: 3683]
    [Request URI: http://localhost:8000/ANZ1.jpg]
    File Data: 142259 bytes
> JPEG File Interchange Format
```

Both ANZ1.jpg and ANZ2.jpg have hidden messages that we can observe in the HxD Hex editor at the bottom of their decoded text.

ANZ1.jpg

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Texto decodificado
00022B00	51	45	14	00	51	45	14	00	51	45	14	00	51	45	14	00	QE..QE..QE..QE..
00022B10	51	45	14	00	51	45	14	00	51	45	14	00	51	45	14	00	QE..QE..QE..QE..
00022B20	51	45	14	00	51	45	14	00	51	45	14	00	51	45	14	00	QE..QE..QE..QE..
00022B30	51	45	14	00	51	45	14	00	51	45	14	00	51	45	14	00	QE..QE..QE..QE..
00022B40	51	45	14	00	51	45	14	00	51	45	14	00	51	45	14	00	QE..QE..QE..QE..
00022B50	51	45	14	00	51	45	14	00	51	45	14	00	51	45	14	00	QE..QE..QE..QE..
00022B60	51	45	14	00	51	45	14	01	FF	D9	59	6F	75	27	76	65	QE..QE..ÿÙYou've
00022B70	20	66	6F	75	6E	64	20	61	20	68	69	64	64	65	6E	20	found a hidden
00022B80	6D	65	73	73	61	67	65	20	69	6E	20	74	68	69	73	20	message in this
00022B90	66	69	6C	65	21	20	49	6E	63	6C	75	64	65	20	69	74	file! Include it
00022BA0	20	69	6E	20	79	6F	75	72	20	77	72	69	74	65	20	75	in your write u
00022BB0	70	2E	0A														p..

ANZ1 message :"You've found a hidden message in this file! Include it in your write"

ANZ2.jpg

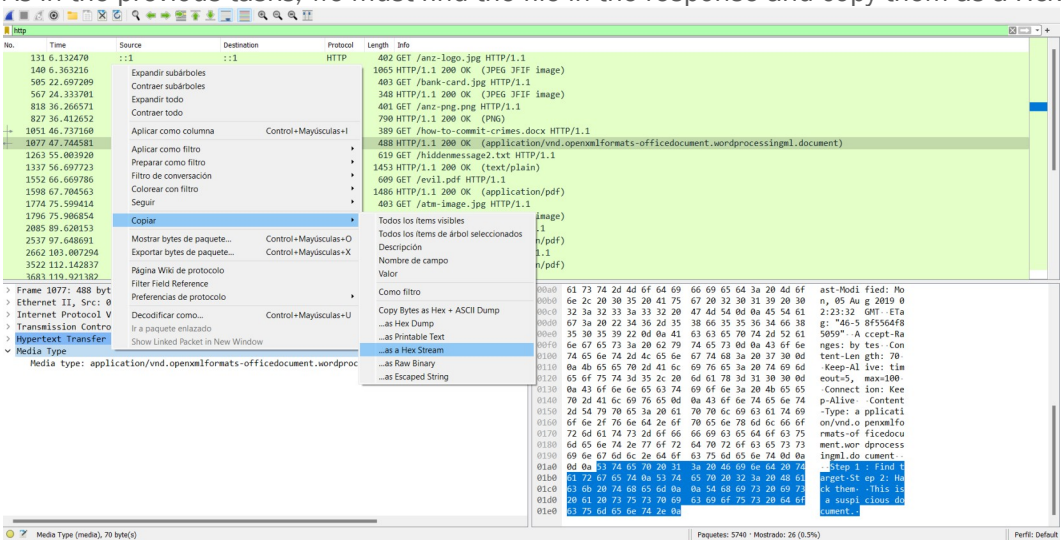
Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Texto decodificado
0002BD00	51	40	05	14	51	40	05	14	51	40	05	14	51	40	05	14	Q@..Q@..Q@..Q@..
0002BD10	51	40	05	14	51	40	05	14	51	40	05	14	51	40	05	14	Q@..Q@..Q@..Q@..
0002BD20	51	40	05	14	51	40	05	14	51	40	05	14	51	40	05	14	Q@..Q@..Q@..Q@..
0002BD30	51	40	05	14	51	40	05	14	51	40	05	14	51	40	1F	FF	Q@..Q@..Q@..Q@.ÿ
0002BD40	D9	59	6F	75	27	76	65	20	66	6F	75	6E	64	20	74	68	ÙYou've found th
0002BD50	65	20	68	69	64	64	65	6E	20	6D	65	73	73	61	67	65	e hidden message
0002BD60	21	0A	49	6D	61	67	65	73	20	61	72	65	20	73	6F	6D	!.Images are som
0002BD70	65	74	69	6D	65	73	20	6D	6F	72	65	20	74	68	61	6E	etimes more than
0002BD80	20	74	68	65	79	20	61	70	70	65	61	72	2E	0A			they appear..

ANZ2 message: "You've found the hidden message!. Images are sometimes more than they appear."

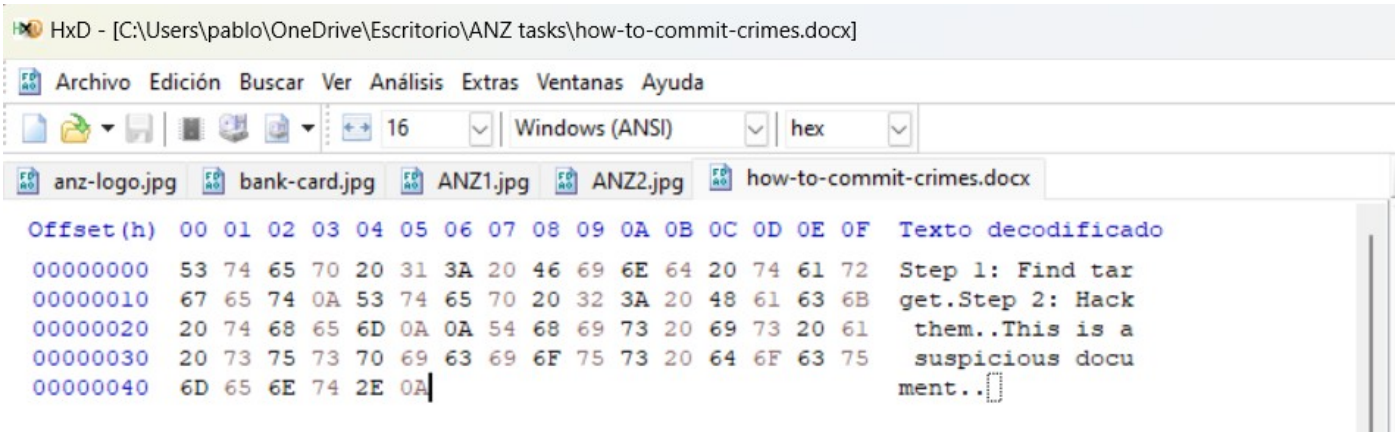
Sub-task 3:

- ***The user downloaded a suspicious document called "how-to-commit-crimes.docx"***
- ***Find the contents of this file and include it in your report.***

As in the previous tasks, we must find the file in the response and copy them as a Hex stream.



We can already see the text in the “decoded text” of the HxD HEX editor or open the file that just saved with any docx reader.



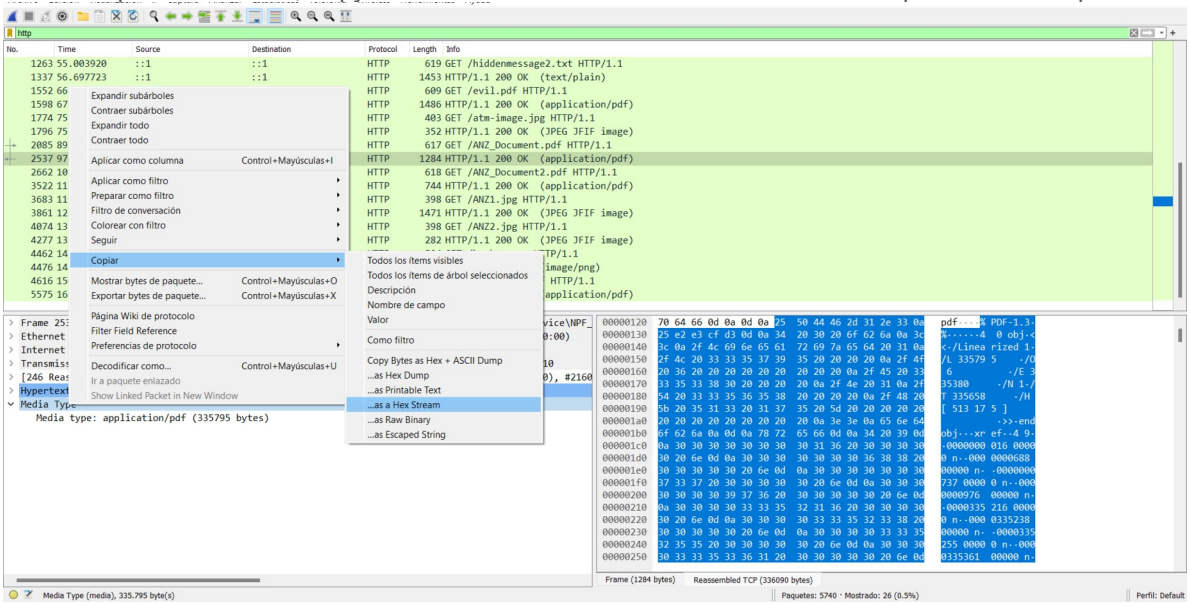
Decoded text:

Step 1: Find target
Step 2: Hack them
This is a suspicious document.

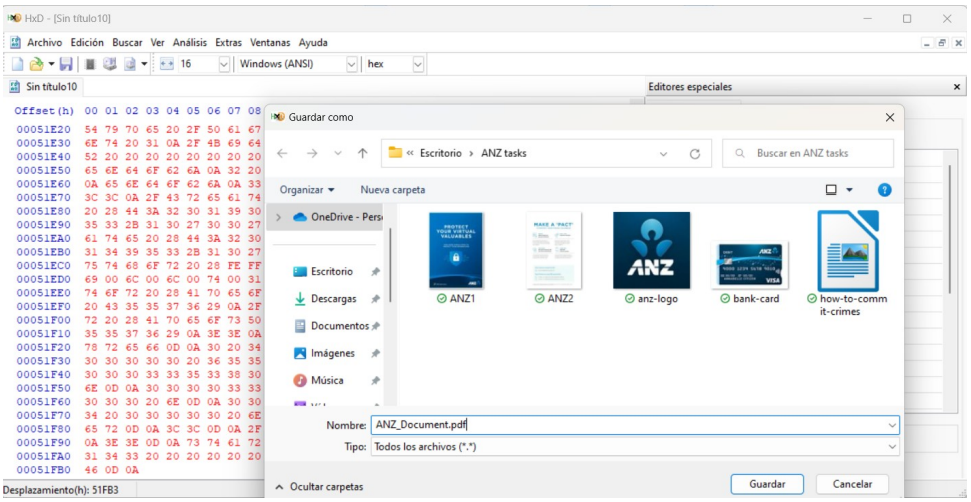
Sub-task 4:

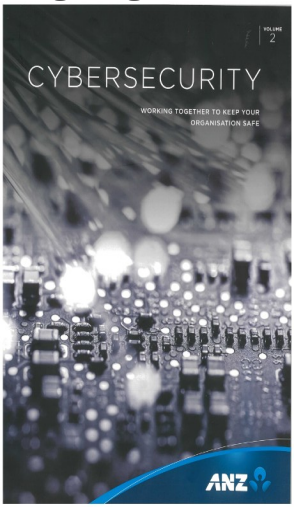
- *The user accessed 3 pdf documents: ANZ_Document.pdf, ANZ_Document2.pdf, evil.pdf*
- *Extract and view these documents. Include images of them in your report.*

As in the previous tasks, copy as HEX stream the code of the file from response to the specific GET request

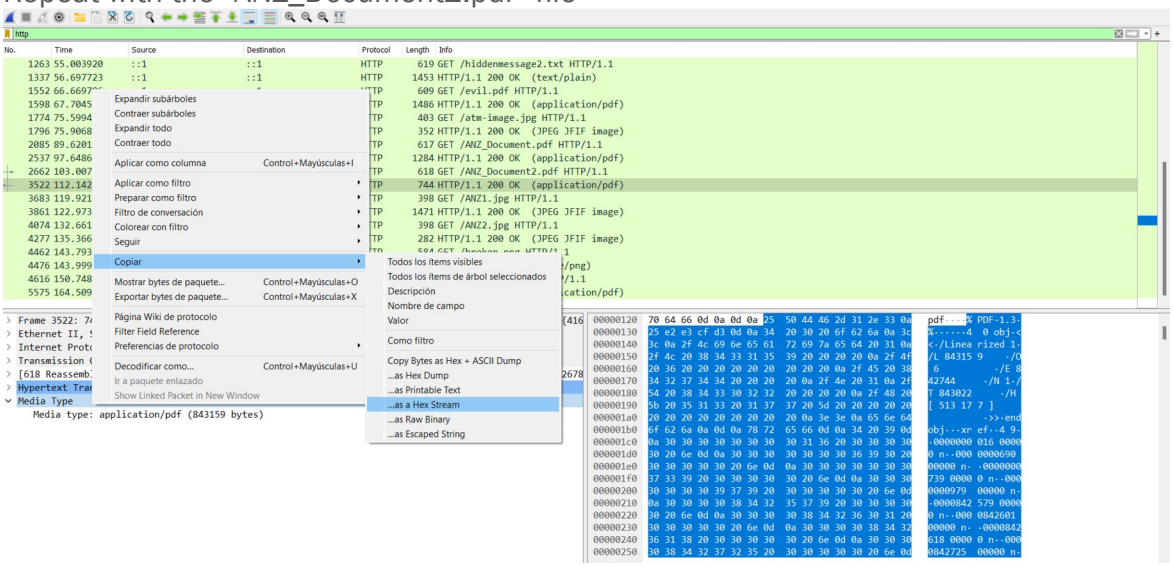


Copy that code in the HxD Hex editor and save the file as pdf obtaining the following image

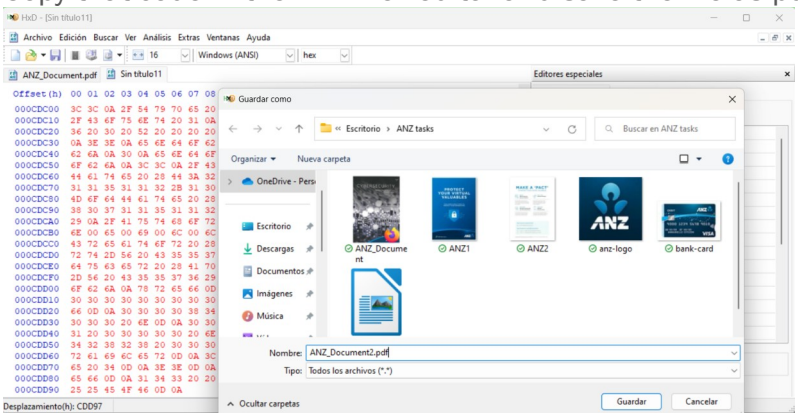


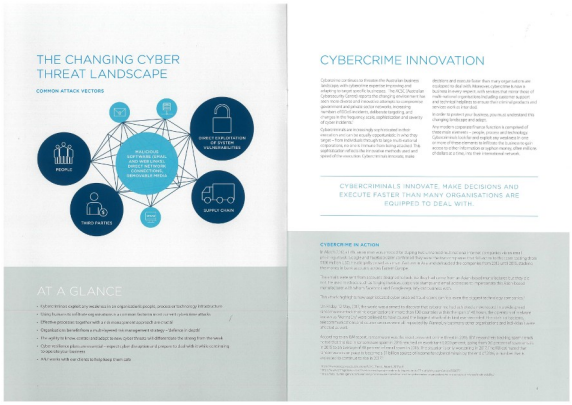


Repeat with the “ANZ_Document2.pdf” file

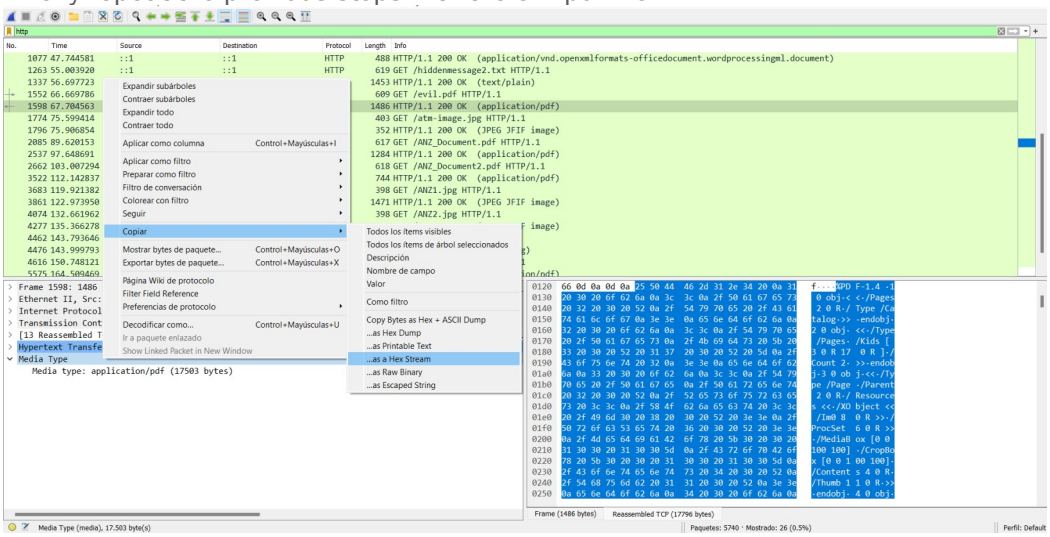


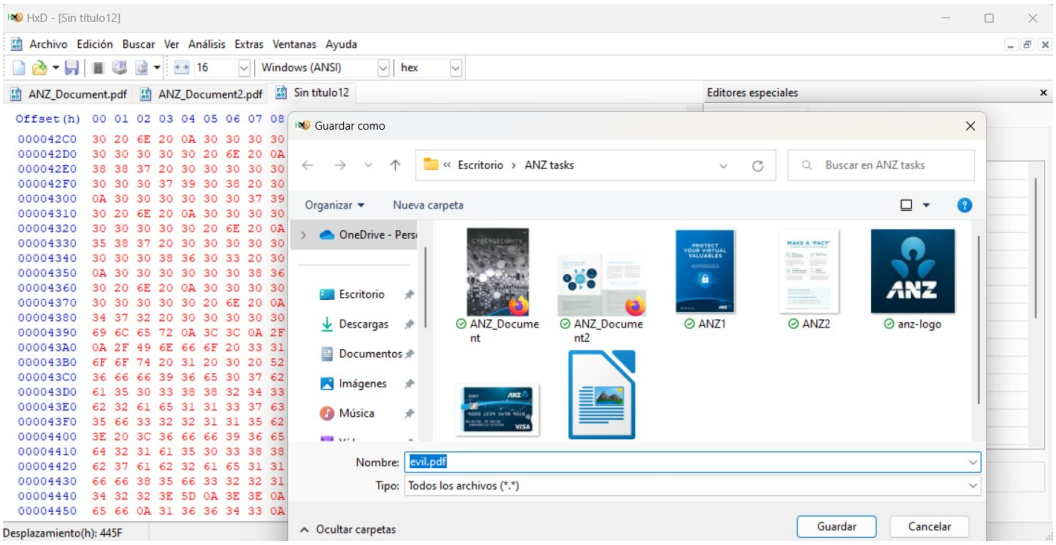
Copy that code in the HxD Hex editor and save the file as pdf obtaining the following file.

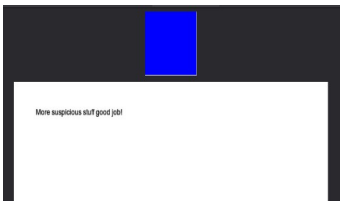




Finally repeat the previous steps with the evil.pdf file.







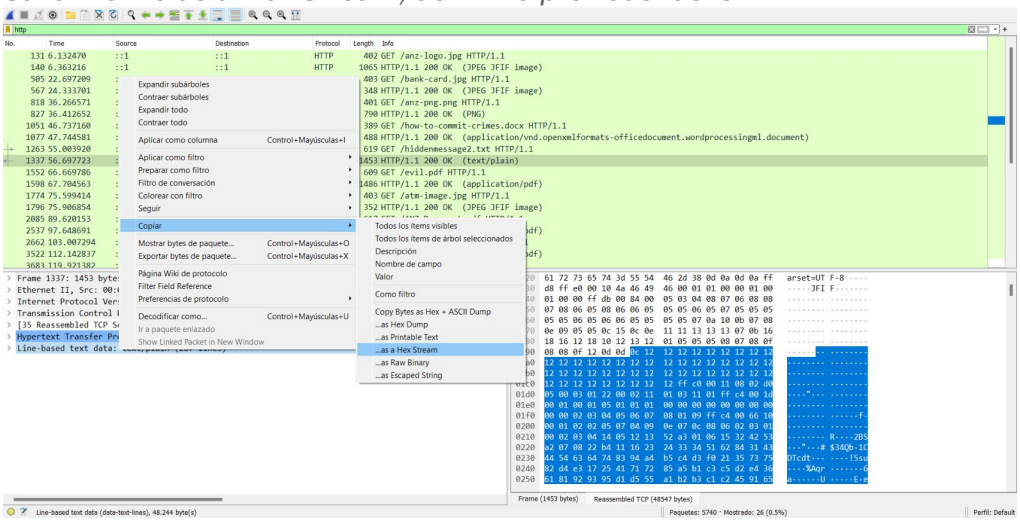
We obtain a 2 pages pdf file that in the first one has a completely blue page and in the second one has the following message:

More suspicious stuff good job!

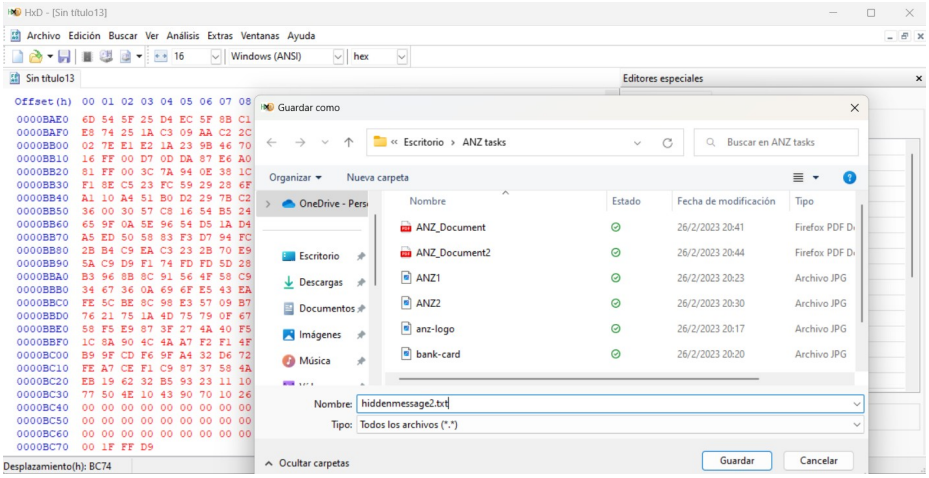
Sub-task 5:

- *The user also accessed a file called "hiddenmessage2.txt"*
- *What is the contents of this file? Include it in your report*

Save the file as a Hex Stream, as in the previous tasks.

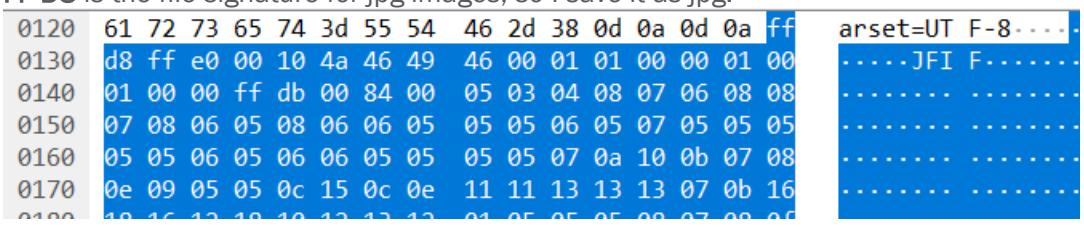


Save it as a text file with the HxD Hex editor

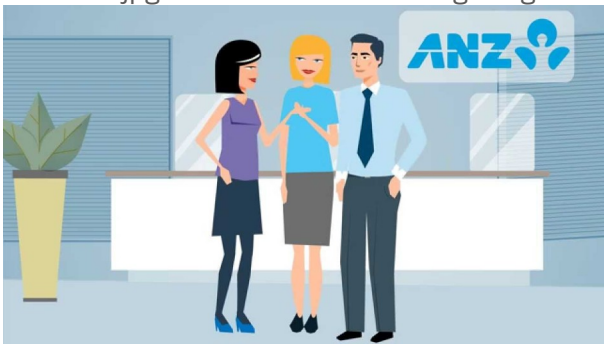


When opening we can notice that the text is impossible to read, but we can see the signature of the JFIF file.

FF D8 is the file signature for jpg images, so I save it as jpg.



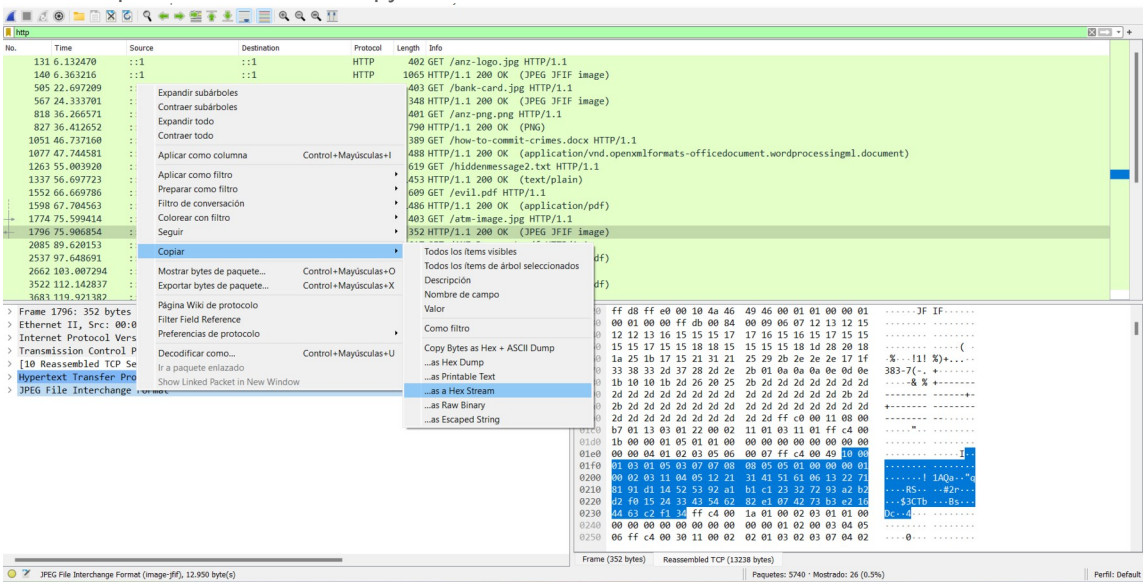
Saved as jpg we can see the following image.



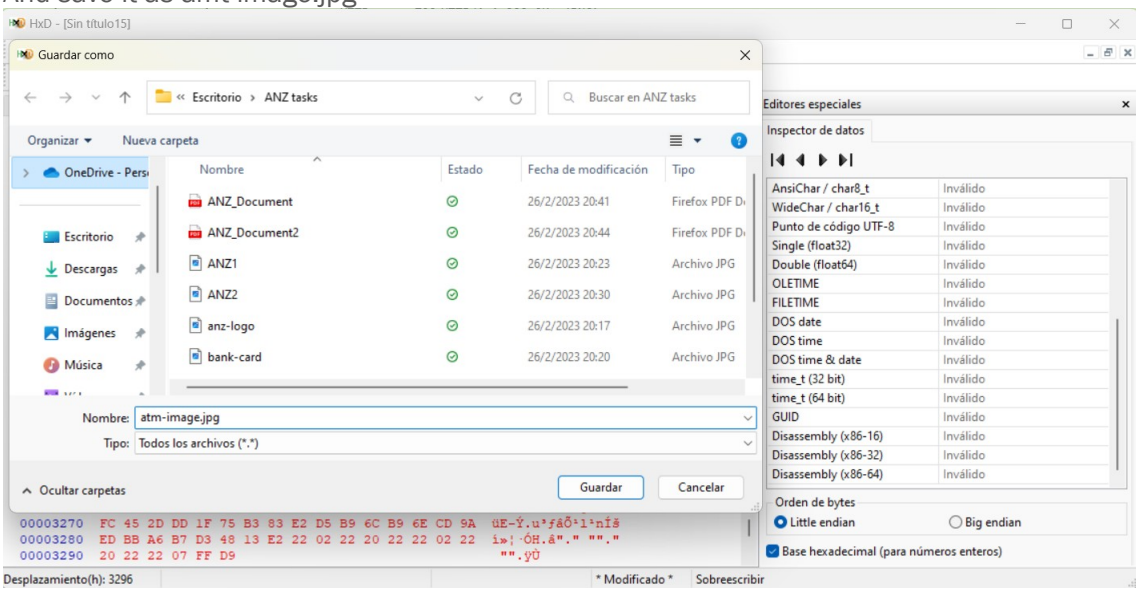
Sub-task 6:

- *The user accessed an image called "atm-image.jpg"*
- *Identify what is different about this traffic and include everything in your report.*

As in the previous tasks we copy the file code as a Hex stream.



And save it as amt-image.jpg



Obtaining the following image.

But wait, there is more!

There is a hidden image embedded in the amt-image.jpg file. That could be noticed when inspecting the file stream with Wireshark. In the image below, I identify the embedded jpg image by its JFIF signature, so I copied its hex code and saved it to a new file called "embedded.jpg" getting the following image of a thief.

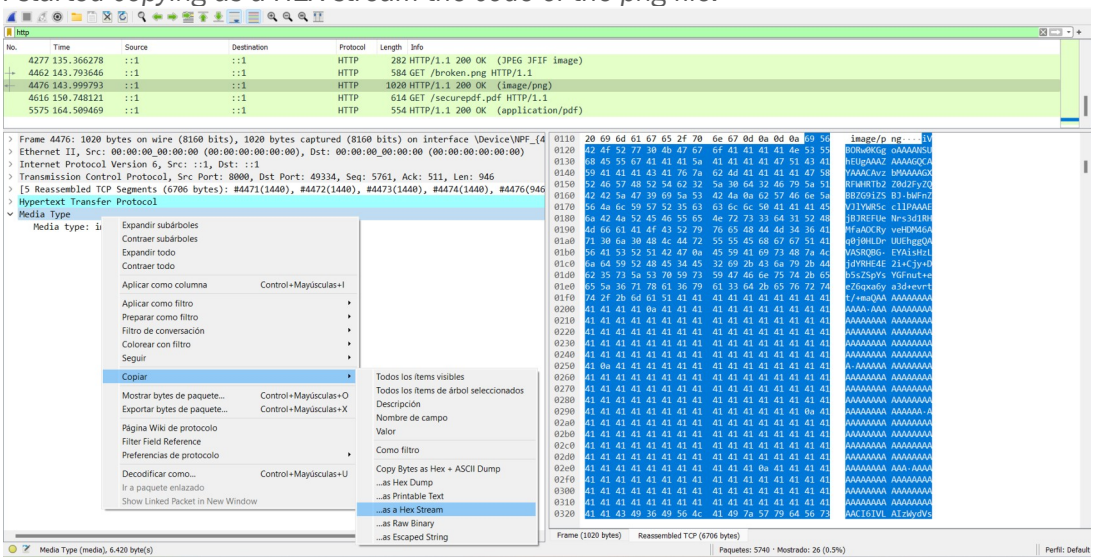
atm-image.jpg

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Texto decodificado
00001E60	8C	02	BA	8B	86	E4	E5	02	23	45	35	D1	3F	0A	6D	12	Œ.º‹†äå.#E5Ñ?.m.
00001E70	83	45	00	2A	6B	9A	A4	19	AE	21	40	91	55	2A	76	04	ƒE.*kš¤.®!@'U*v.
00001E80	8A	10	C4	31	4E	D7	2E	5C	A8	43	96	10	82	23	C5	DC	Š.Ä1N×.\¨C–.‚#ÅÜ
00001E90	81	32	ED	5C	B9	05	D4	69	74	42	19	13	0A	E5	C9	C4	.2i\¹.ÔitB...åÉÄ
00001EA0	35	3C	90	B6	67	80	AE	E5	9D	93	09	6B	86	84	AE	5C	5<.¶g€®å.".k†„®\
00001EB0	A9	E9	33	47	5C	66	79	8C	52	96	8A	E5	A7	1C	BD	8B	©é3G\fyŒR–Šå§.½‹
00001EC0	97	2D	28	CC	28	6A	92	20	B9	72	20	25	C2	A5	60	48	—-(Ì(j' ¹r %Â¥`H
00001ED0	B9	10	13	34	28	1C	E1	53	99	5C	B9	40	0E	1D	AB	85	¹..4(.áS™\¹@..«…
00001EE0	12	2E	50	82	87	22	8B	8D	34	A9	5C	B9	42	33	98	0E	..P‚‡"‹.4©\¹B3˜.
00001EF0	DC	94	AC	62	45	C8	80	7B	B2	A3	B7	7C	77	AE	96	D0	Ü"¬bEÈ€{²£·|w®–Ð
00001F00	5B	95	41	35	22	94	35	00	68	49	D3	FF	00	8B	97	20	[•A5"”5.hIÓÿ.‹—
00001F10	42	27	DB	0E	E0	A4	B2	BB	2A	6E	F7	6C	48	B9	40	A2	B'Û.à¤²»*n÷lH¹@¢
00001F20	7A	25	09	17	22	11	1C	07	F7	4D	C5	EC	5C	B9	00	8A	z%..".....÷MÅì\¹.Š
00001F30	D7	AE	24	AE	5C	A1	0E	0E	29	CD	90	15	CB	94	20	B5	×®$®\¡..)Í..Ë” µ
00001F40	5C	B9	72	84	3F	FF	D9	FF	D8	FF	E0	00	10	4A	46	49	\¹r„?ÿÙÿØÿà..JFI
00001F50	46	00	01	01	00	00	01	00	01	00	00	FF	DB	00	84	00	F.....ÿÛ.„.
00001F60	09	06	07	12	13	12	15	12	12	12	16	15	15	15	18	17	.....
00001F70	17	15	16	15	15	15	16	16	18	1B	17	15	16	17	17	19	.....
00001F80	15	13	18	1E	2A	22	18	1F	26	1B	16	15	21	31	21	26	....*"..&...!1!&
00001F90	29	2B	2E	2E	2F	18	1F	33	38	33	2C	37	28	2D	2E	2B	)+../..383,7(-.+
00001FA0	01	0A	0A	0A	0D	0D	0E	15	0F	10	17	2B	19	17	19	2D	.....+...-
00001FB0	2B	2B	37	2B	2D	2B	2D	2B	2D	37	2B	2D	2D	38	2D	2B	++7+-+-+-7+--8-+
00001FC0	30	37	38	2E	2D	2D	38	2D	2B	2D	2B	37	2B	2B	2B	37	078.--8-+-+7+++7
00001FD0	2D	2B	37	2B	2B	2B	2B	2B	2D	2B	37	2D	37	2B	2B	2B	-+7+++++-+7-7+++

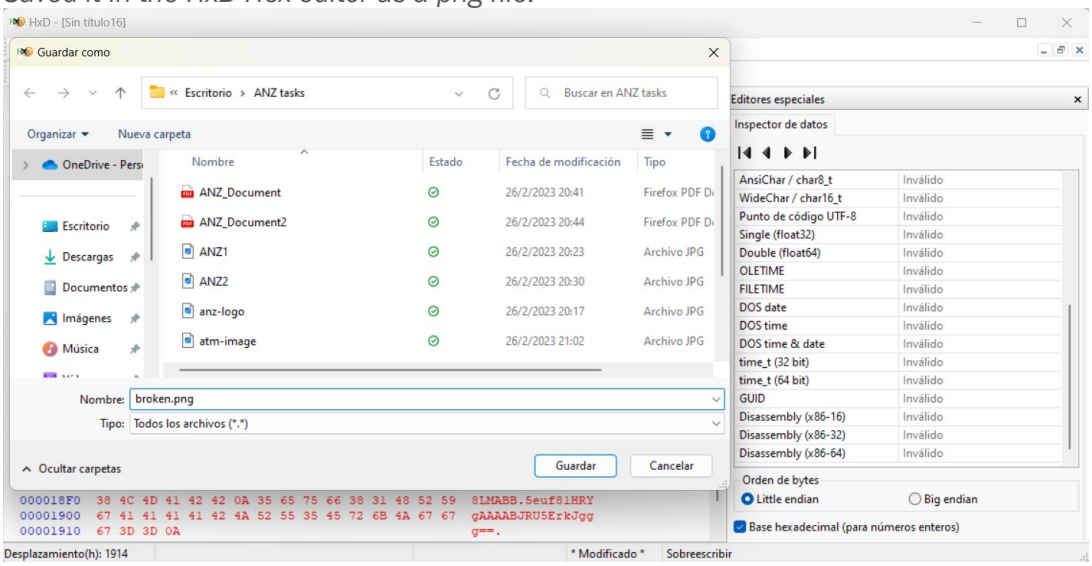
Sub-task 7:

- *The network traffic shows that the user accessed the image "broken.png"*
- *Extract and include the image in your report.*

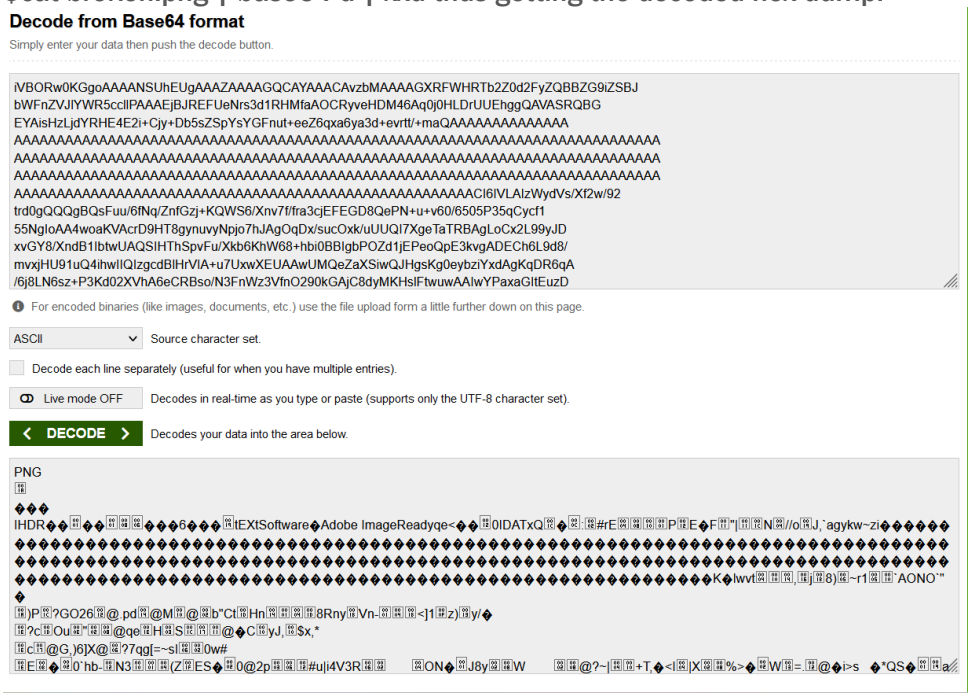
I started copying as a HEX stream the code of the png file.



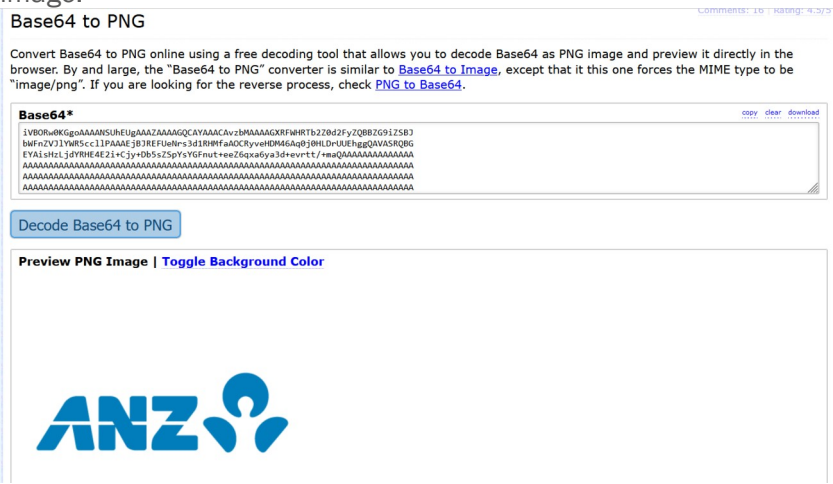
Saved it in the HxD Hex editor as a png file.



I couldn't open the file or identify the file signature, so I tried to investigate if the file was encrypted. As we can see with the online decoder https://www.base64decode.org/ the broken.png file is encoded in Base64 and once decoded we can already read some information about it. We can also go to the terminal and type:
$cat broken.png | base64-d | xxd thus getting the decoded hex dump.

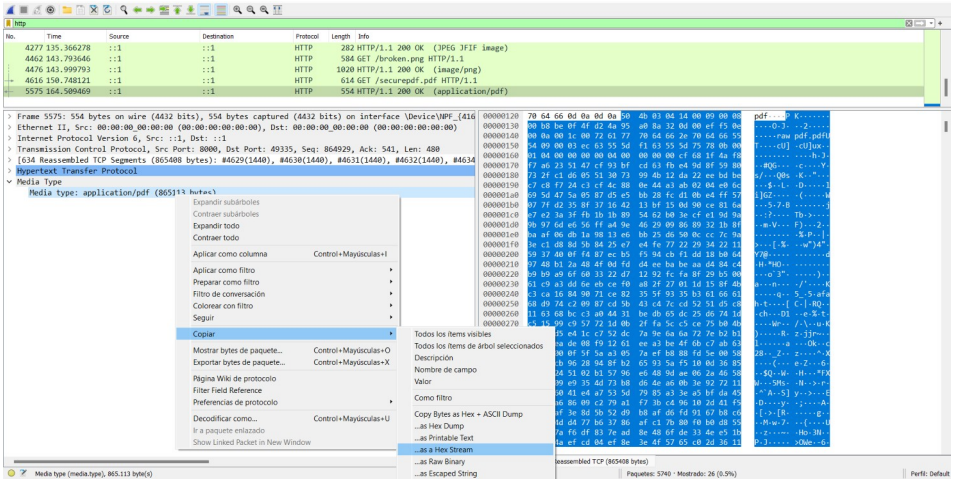


Now that we know the file is Base64 encoded, we can go to the terminal and redirect the contents of broken.png to a new file or use online tools like https://base64.guru/converter/decode/image/png to decode and get the png image.

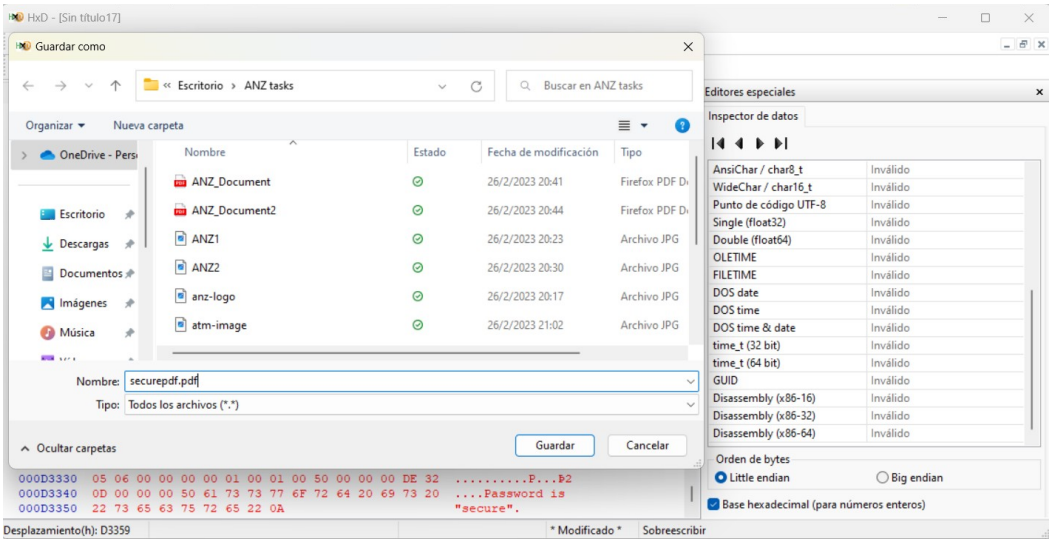


Sub-task 8:

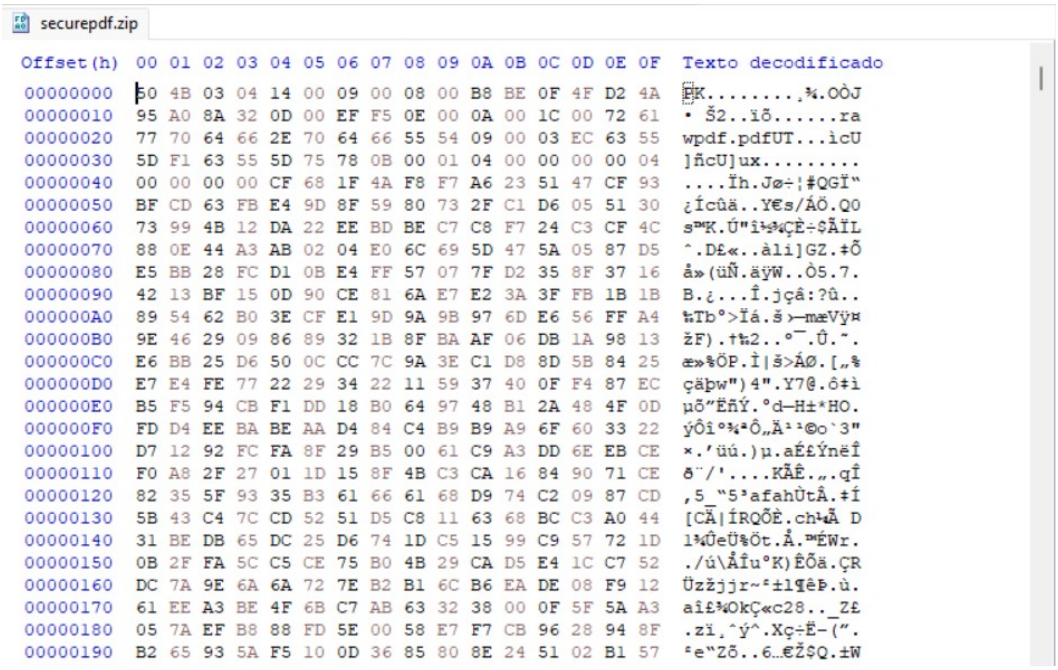
- *The user accessed one more document called securepdf.pdf*
- *Access this document include an image of the pdf in your report. Detail the steps to access it.*



I tried to save it as a pdf file first as in the previous tasks but it wasn't working.

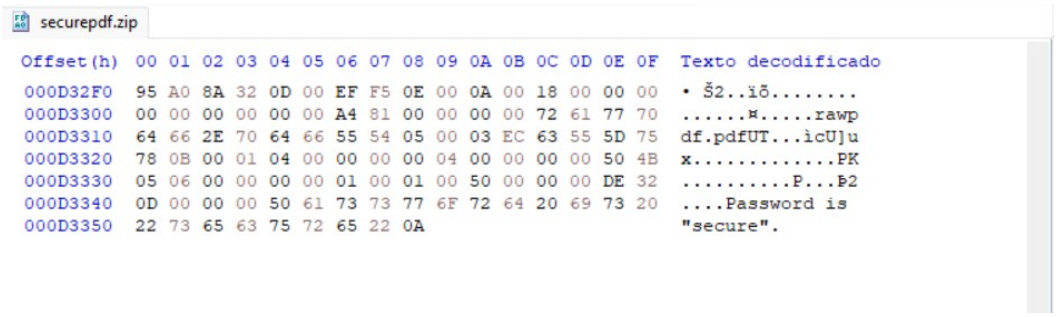


Then I noticed that the file signature (50 4B 03 04) indicates that it was a zip file instead of a pdf. So I save it as securepdf.zip


securepdf.zip

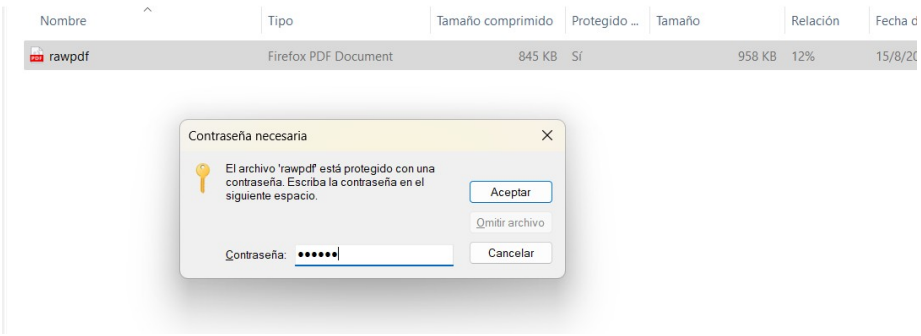
Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Texto decodificado
00000000	50	4B	03	04	14	00	09	00	08	00	B8	BE	0F	4F	D2	4A	PK.....¸¾.OÒJ
00000010	95	A0	8A	32	0D	00	EF	F5	0E	00	0A	00	1C	00	72	61	• Š2..ïõ.....ra
00000020	77	70	64	66	2E	70	64	66	55	54	09	00	03	EC	63	55	wpdf.pdfUT...ìcU
00000030	5D	F1	63	55	5D	75	78	0B	00	01	04	00	00	00	00	04	]ñcU]ux.....
00000040	00	00	00	00	CF	68	1F	4A	F8	F7	A6	23	51	47	CF	93	....Ïh.Jø÷¦#QGÏ"
00000050	BF	CD	63	FB	E4	9D	8F	59	80	73	2F	C1	D6	05	51	30	¿Ícûä..Y€s/ÁÖ.Q0
00000060	73	99	4B	12	DA	22	EE	BD	BE	C7	C8	F7	24	C3	CF	4C	s™K.Ú"i½¾ÇÈ÷$ÃÏL
00000070	88	0E	44	A3	AB	02	04	E0	6C	69	5D	47	5A	05	87	D5	^.D£«..àli]GZ.‡Õ
00000080	E5	BB	28	FC	D1	0B	E4	FF	57	07	7F	D2	35	8F	37	16	å»(üÑ.äÿW..Ò5.7.
00000090	42	13	BF	15	0D	90	CE	81	6A	E7	E2	3A	3F	FB	1B	1B	B.¿...Î.jçâ:?û..
000000A0	89	54	62	B0	3E	CF	E1	9D	9A	9B	97	6D	E6	56	FF	A4	‰Tb°>Ïá.š›—mæVÿ¤
000000B0	9E	46	29	09	86	89	32	1B	8F	BA	AF	06	DB	1A	98	13	žF).†‰2..º¯.Û.˜.
000000C0	E6	BB	25	D6	50	0C	CC	7C	9A	3E	C1	D8	8D	5B	84	25	æ»%ÖP.Ì|š>ÁØ.[„%
000000D0	E7	E4	FE	77	22	29	34	22	11	59	37	40	0F	F4	87	EC	çäþw")4".Y7@.ô‡ì
000000E0	B5	F5	94	CB	F1	DD	18	B0	64	97	48	B1	2A	48	4F	0D	µõ"ËñÝ.°d—H±*HO.
000000F0	FD	D4	EE	BA	BE	AA	D4	84	C4	B9	B9	A9	6F	60	33	22	ýÔî°¾ªÔ„Ä¹¹©o`3"
00000100	D7	12	92	FC	FA	8F	29	B5	00	61	C9	A3	DD	6E	EB	CE	×.'üú.)µ.aÉ£ÝnëÎ
00000110	F0	A8	2F	27	01	1D	15	8F	4B	C3	CA	16	84	90	71	CE	ð¨/'....KÃÊ.„.qÎ
00000120	82	35	5F	93	35	B3	61	66	61	68	D9	74	C2	09	87	CD	‚5_"5³afahÙtÂ.‡Í
00000130	5B	43	C4	7C	CD	52	51	D5	C8	11	63	68	BC	C3	A0	44	[CÄ|ÍRQÕÈ.ch¼Ã D
00000140	31	BE	DB	65	DC	25	D6	74	1D	C5	15	99	C9	57	72	1D	1¾ÛeÜ%Öt.Å.™ÉWr.
00000150	0B	2F	FA	5C	C5	CE	75	B0	4B	29	CA	D5	E4	1C	C7	52	./ú\ÅÎu°K)ÊÕä.ÇR
00000160	DC	7A	9E	6A	6A	72	7E	B2	B1	6C	B6	EA	DE	08	F9	12	Üzžjjr~²±l¶êÞ.ù.
00000170	61	EE	A3	BE	4F	6B	C7	AB	63	32	38	00	0F	5F	5A	A3	aî£¾OkÇ«c28.._Z£
00000180	05	7A	EF	B8	88	FD	5E	00	58	E7	F7	CB	96	28	94	8F	.zï¸ˆý^.Xç÷Ë–(".
00000190	B2	65	93	5A	F5	10	0D	36	85	80	8E	24	51	02	B1	57	²e"Zõ..6…€Ž$Q.±W

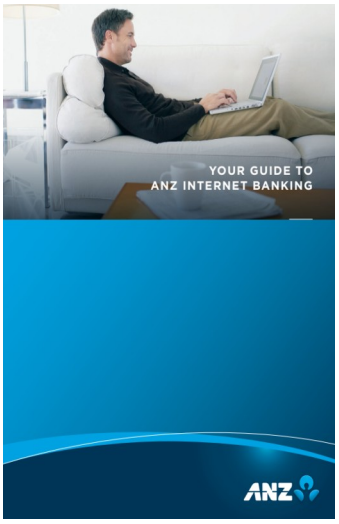
At the end of the decoded text we can see that the password to open the content in the zip folder is “secure”, so we proceed to open the file.


securepdf.zip

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Texto decodificado
000D32F0	95	A0	8A	32	0D	00	EF	F5	0E	00	0A	00	18	00	00	00	• Š2..ïõ.....
000D3300	00	00	00	00	00	00	A4	81	00	00	00	00	72	61	77	70	......¤.....rawp
000D3310	64	66	2E	70	64	66	55	54	05	00	03	EC	63	55	5D	75	df.pdfUT...ìcU]u
000D3320	78	0B	00	01	04	00	00	00	00	04	00	00	00	00	50	4B	x.............PK
000D3330	05	06	00	00	00	00	01	00	01	00	50	00	00	00	DE	32	..........P...Þ2
000D3340	0D	00	00	00	50	61	73	73	77	6F	72	64	20	69	73	20	....Password is
000D3350	22	73	65	63	75	72	65	22	0A								"secure".

With the “secure” password we can open the following pdf file





Links used for the resolution:

https://en.wikipedia.org/wiki/List_of_file_signatures
https://www.base64decode.org/
https://base64.guru/converter/decode/image/png

Sub-task 7 commands explained:
$cat broken.png | base64 -d | xxd
cat : Concatenates FILE(s), or standard input, to standard output.
Base64 -d : Decodes the Base64 code.
Xxd : creates a hex dump of a given file or standard input.