Packet Capture Analysis:

Given the context, we know the user has viewed and accessed various images and files. I started by investigating the PCAP for HTTP GET requests, as these indicate the user requesting resources.

http.request.method==GET

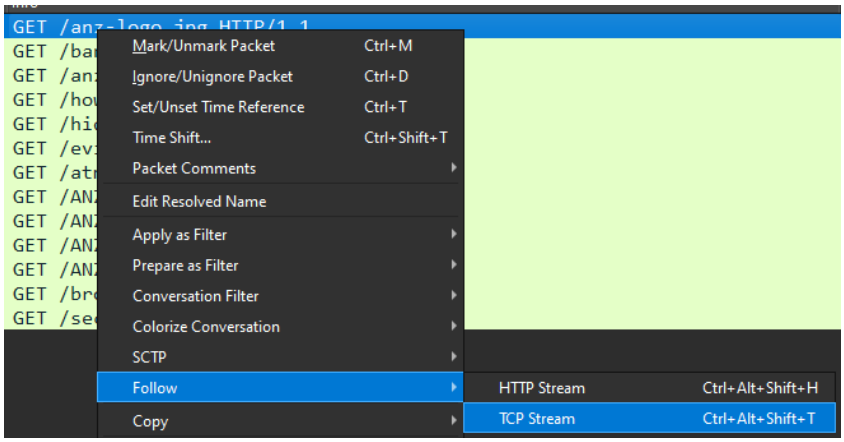
No.	Time	Source	Destination	Destination Port	Protocol	Info
131	2019-08-16 00:47:40	::1	::1	8000	HTTP	GET /anz-logo.jpg HTTP/1.1
505	2019-08-16 00:47:56	::1	::1	8000	HTTP	GET /bank-card.jpg HTTP/1.1
818	2019-08-16 00:48:10	::1	::1	8000	HTTP	GET /anz-png.png HTTP/1.1
1051	2019-08-16 00:48:21	::1	::1	8000	HTTP	GET /how-to-commit-crimes.docx HTTP/1.1
1263	2019-08-16 00:48:29	::1	::1	8000	HTTP	GET /hiddenmessage2.txt HTTP/1.1
1552	2019-08-16 00:48:40	::1	::1	8000	HTTP	GET /evil.pdf HTTP/1.1
1774	2019-08-16 00:48:49	::1	::1	8000	HTTP	GET /atm-image.jpg HTTP/1.1
2085	2019-08-16 00:49:03	::1	::1	8000	HTTP	GET /ANZ_Document.pdf HTTP/1.1
2662	2019-08-16 00:49:17	::1	::1	8000	HTTP	GET /ANZ_Document2.pdf HTTP/1.1
3683	2019-08-16 00:49:34	::1	::1	8000	HTTP	GET /ANZ1.jpg HTTP/1.1
4074	2019-08-16 00:49:46	::1	::1	8000	HTTP	GET /ANZ2.jpg HTTP/1.1
4462	2019-08-16 00:49:58	::1	::1	8000	HTTP	GET /broken.png HTTP/1.1
4616	2019-08-16 00:50:05	::1	::1	8000	HTTP	GET /securepdf.pdf HTTP/1.1

From the above display filter, we can see that the user accessed a series of documents and images.

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.*

To extract these images manually, right-click the relevant packet and select “Follow TCP Stream.”

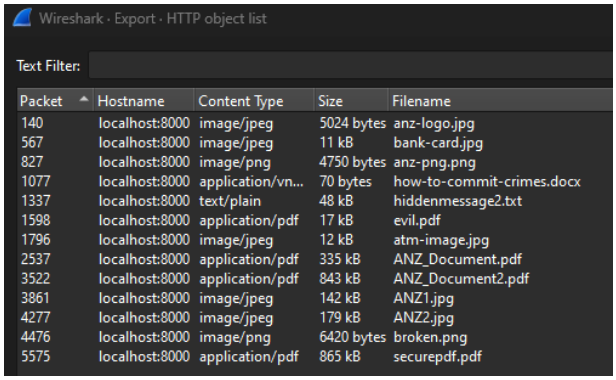


Change the encoding from ASCII to raw. Identify the JPEG file by searching for the hex header FFD8 and footer FFD9. Copy this hex range into HxD to reconstruct and save the images.

```
FF D8 FF E0 00 10 4A 46 49 46 00 01 01 00 00 01  ÿØÿà..JFIF......
00 01 00 00 FF DB 00 84 00 09 06 07 0F 0F 0F 0D  ....ÿÛ.„........
0F 0F 0F 10 0F 0F 0F 0F 0F 10 0F 0D 0F 0D 0F 0F  ................
10 0E 10 15 11 16 16 15 15 16 15 18 1D 28 21 18  .............(!.
1A 25 1B 15 15 21 32 21 25 29 2B 2E 2E 30 17 20  .%...!2!%)+..0.
3F 44 33 2D 37 28 2D 2E 2B 01 0A 0A 0A 0E 0D 0E  ?D3-7(-.+.......
1A 10 10 17 2B 26 1E 22 2D 2D 2F 2F 2B 2D 2D 2D  ....+&."--//+---
```

If you follow the same process for bank-card.jpg, we are left with the following image:

Alternatively, you can simply navigate to File > Export Objects > HTTP:



However, given the context surrounding this question, we are meant to carve out the file manually.

Sub-task 2:

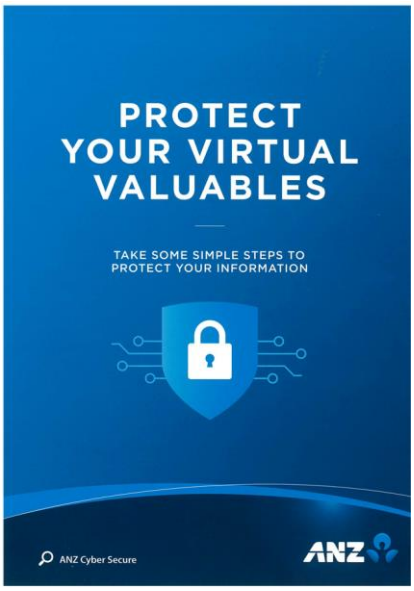
- *The network traffic for the images "ANZ1.jpg" and "ANZ2.jpg" is more than it appears.*
- *Extract the images, include them and mention what is different about them in your report.*

As evident in the pcap, ANZ1 and ANZ2 were downloaded in quick succession:

```
2019-08-16 00:49:34   ::1                ::1                8000       HTTP    GET /ANZ1.jpg HTTP/1.1
2019-08-16 00:49:46   ::1                ::1                8000       HTTP    GET /ANZ2.jpg HTTP/1.1
```

After carving out these files following the same process as subtask 1, they appear to be part of the same document:

ANZ1.jpg:



ANZ2.jpg:

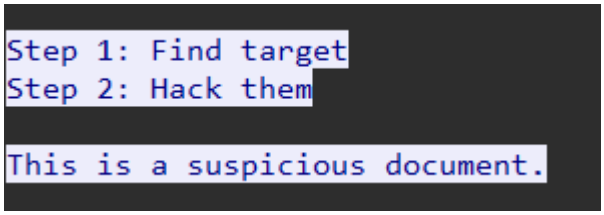
Looking at both documents in HxD, we can see that at the end of the ANZ2.jpg image is a secret message:

```
ÙYou've found th
e hidden message
!.Images are som
etimes more than
 they appear..
```

Sub-task 3:

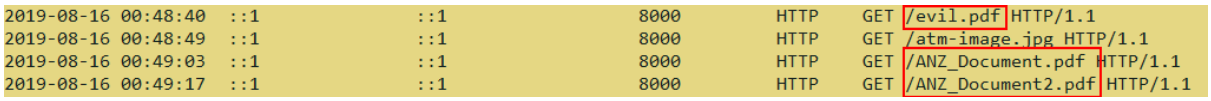
- *The user downloaded a suspicious document called "how-to-commit-crimes.docx"*
- *Find the contents of this file and include it in your report.*

The document content can be accessed via the TCP stream of the relevant packet.



Sub-task 4:

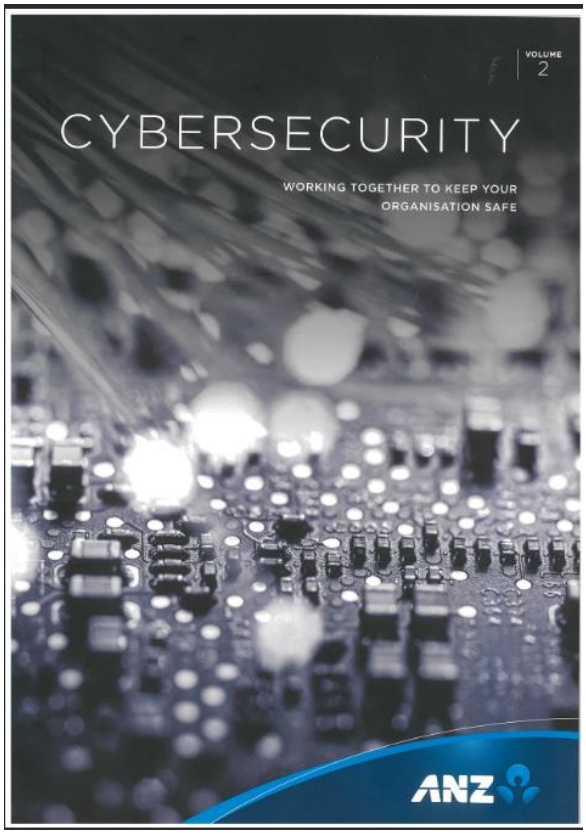
- *The user accessed 3 pdf documents: ANZ_Document.pdf, ANZ_Document2.pdf, evil.pdf*
- *Extract and view these documents. Include images of them in your report.*



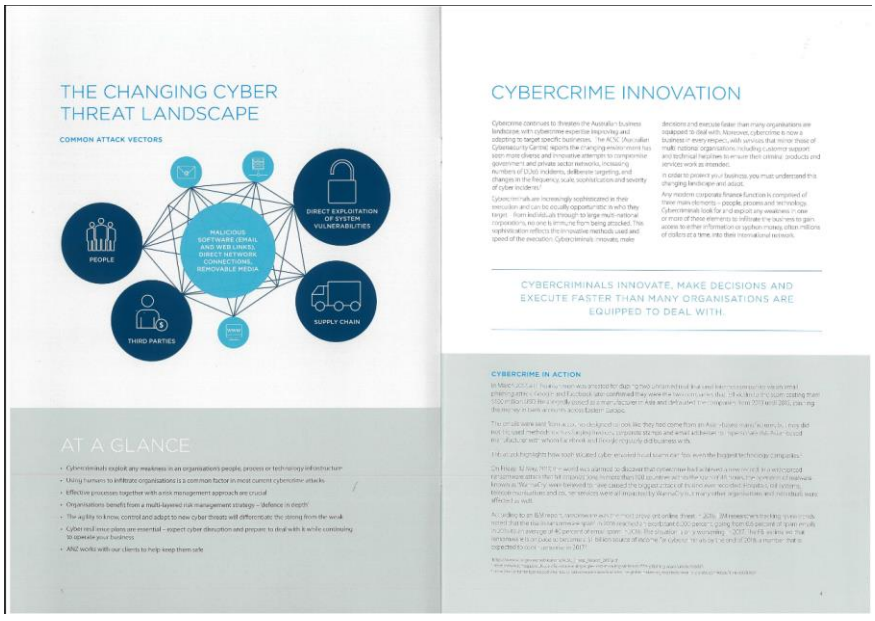
Starting with evil.pdf, in order to carve out the pdf file from the TCP stream you need to copy the hex from 25 50 44 46 2D to 25 25 45 4F 46. After pasting this hex into HxD and saving it as a pdf, you can view the file:

After following the same process for the other 2 PDFs, you are presented with:

ANZ_Document:



ANZ_Document2:



Sub-task 5:

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

```
2019-08-16 00:48:29  ::1              ::1                  8000        HTTP    GET /hiddenmessage2.txt HTTP/1.1
```

If you view the TCP stream for this packet, you can see based on the header that this is a jpeg image, and not a TXT document:

```
HTTP/1.1 200 OK
Date: Fri, 16 Aug 2019 00:48:25 GMT
Server: Apache/2.4.6 (CentOS)
Last-Modified: Fri, 09 Aug 2019 04:47:39 GMT
ETag: "bc74-58fa7dfb63089"
Accept-Ranges: bytes
Content-Length: 48244
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/plain; charset=UTF-8

......JFIF.....................................................
.....    ..................................
```

If you follow the same process as subtask one, you will be given the following image:

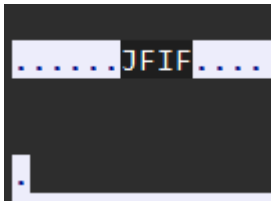
After viewing the file, I cannot identify any hidden content.

Sub-task 6:

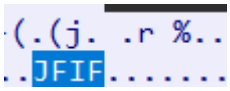
- *The user accessed an image called "atm-image.jpg"*
- *Identify what is different about this traffic and include everything in your report.*

```
2019-08-16 00:48:49  ::1              ::1                8000        HTTP    GET /atm-image.jpg HTTP/1.1
```

If you follow the TCP stream of this packet, it reveals two JPEG headers, indicating two embedded images within the single TCP stream.



And



Let's carve out both by copying the hex between FFD8 and FFD9. The first image looks like as follows:

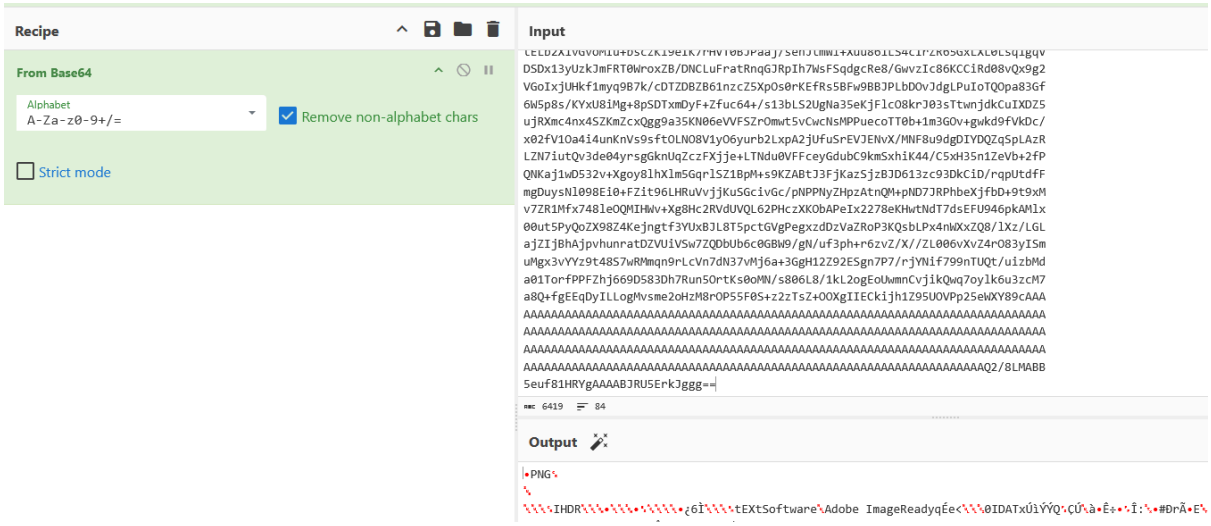
And the second image is as follows:

Sub-task 7:

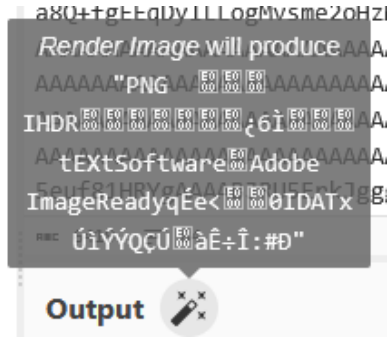
- *The network traffic shows that the user accessed the image "broken.png"*
- *Extract and include the image in your report.*

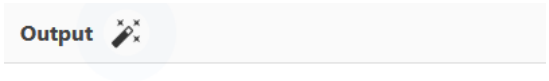
```
2019-08-16 00:49:58  ::1              ::1                  8000        HTTP    GET /broken.png HTTP/1.1
```

If you follow the TCP stream of this packet, you can immediately assume that this content is encoded in what appears to be base64. To test this theory, let's decode this text using CyberChef and try to save it as a png file:

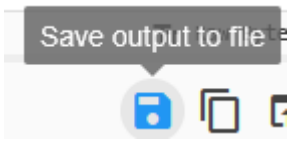


CyberChef is so feature rich that it can even render the image for you:





Alternatively, you can save the output to a png file and view it normally.



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.*

```
2019-08-16 00:50:05  ::1              ::1                   8000        HTTP    GET /securepdf.pdf
```

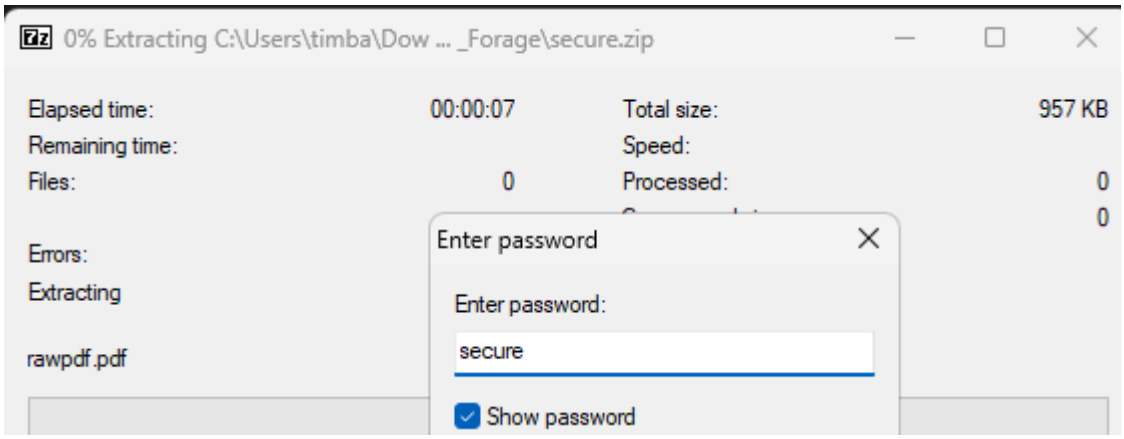
After taking a look at the raw text, it becomes clear that this is a ZIP file, and not a PDF. This is evident by the PK ASCII at the start:

```
PK....   ......O.J...2
```

At the end of the text is also what appears to be a password, likely for a password protected zip file.

```
.................rawpdf.pdfUT....cU]ux.............PK..........P....2
...Password is "secure"
```

To carve out the file, copy the hex from 504B0304 to 504b0506, and save it as a zip file. Then unzip it using a tool like 7z and provide the password “secure”:



You are then given the following PDF called rawpdf.pdf:

YOUR GUIDE TO ANZ INTERNET BANKING

TABLE OF CONTENTS