蝎

Try Hack Me

Confidential

We got our hands on a confidential case file from some self-declared "black hat hackers"... it looks like they have a secre...

security forensics pdf qr Easy

sc0rp10n [0xD][God]

158 355 23

tryhackme.com

This was a really easy but interesting challenge, as I had to study again some stuff about image formats, composition and editing.

First, I transferred the file from the challenge machine into my machine.

You can do it by spawning a simple python http server from the challenge's machine

```
python3 -m http.server 9090
```

Apart from reading the contents of the file, the challenge's description talked about a QR Code inside the file, so my attention was immediately caught by the QR.

This is the redacted QR

The goal is to read the QR code to retrieve the flag, but we need to get rid of the alert symbol

At first, I tried using Binwalk to see if something would come up, but nothing.

Then, I immediately went to check the metadata of the file.

This can be done using exiftool.

```
~/Desktop/CTF/Confidential  exiftool Repdf.pdf
ExifTool Version Number         : 12.44
File Name                       : Repdf.pdf
Directory                       : .
File Size                       : 103 kB
File Modification Date/Time     : 2022:03:11 15:56:29-05:00
File Access Date/Time           : 2022:08:22 15:46:14-05:00
File Inode Change Date/Time     : 2022:08:22 15:00:37-05:00
File Permissions                : -rw-r--r--
File Type                       : PDF
File Type Extension             : pdf
MIME Type                       : application/pdf
PDF Version                     : 1.5
Linearized                      : No
Page Count                      : 1
Producer                        : cairo 1.17.4 (https://cairographics.org)
Create Date                     : 2022:02:05 18:31:35+05:30
~/Desktop/CTF/Confidential
```

Every header looks normal, except for the **“Producer”** header. By saying this, I do not mean that it is unusual, but it gave me a hint on where to look.

I went to google Cairo 1.17.4 to see what will come up. I found about a graphics library called Cairographics, so I looked up if there was any type of CLI tool for linux and I found it.

There's this tool called pdftocairo that lets you transform Cairo pdf generated files to some other kind of image formats.

After installing the tool with apt, trying to use it on the file showed me this. The tool is able to transform the pdf file into any of those formats, I just had to place the argument to transform it.

If you know a little bit about some image formats, you are going to think as I do that SVG is our best option here

SVG stands for **Scalable Vector Graphics**, you can go here to read more about it.

https://en.wikipedia.org/wiki/Scalable_Vector_Graphics

```
~/Desktop/CTF/Confidential  pdftocairo Repdf.pdf -svg
~/Desktop/CTF/Confidential  ls -aglo
total 316
drwxr-xr-x  2   4096 Aug 22 16:09 .
drwxr-xr-x 41   4096 Aug 22 14:43 ..
-rw-r--r--  1 102818 Mar 11 15:56 Repdf.pdf
-rw-r--r--  1 207051 Aug 22 16:09 Repdf.svg
~/Desktop/CTF/Confidential  file Repdf.svg
Repdf.svg: SVG Scalable Vector Graphics image
~/Desktop/CTF/Confidential
```

Great! The tool successfully transformed the file into an SVG format.

Long story short, you can now open the SVG file in a Browser and edit the file like an HTML page!

```
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="849.893763pt"
height="1099.862517pt" viewBox="0 0 849.893763 1099.862517" version="1.2"> scroll overflow
▶ <defs>…</defs>
▼ <g id="surface1">
  ▶ <g clip-path="url(#clip1)" clip-rule="nonzero">…</g>
  ▼ <g clip-path="url(#clip2)" clip-rule="nonzero">
    ▼ <use xlink:href="#surface14" transform="matrix(1,0,0,1,325,561)">
      ▼ #shadow-root (closed)
        ▼ <g id="surface14" clip-path="url(#clip3)">
          ▼ <g clip-path="url(#clip4)" clip-rule="nonzero">
            ▼ <use xlink:href="#image10" mask="url(#mask0)"
              transform="matrix(0.99975,0,0,0.99975,0.918515,0.859525)">
              ▶ #shadow-root (closed)
              </use>
            </g>
          </g>
      </use>
    </g>
  </g>
</svg>
```

Unraveling the file using the Developer Tools of Firefox, I found a tag that referred to something called **image10.**

```
▼<use xlink:href="#image10" mask="url(#mask0)"
  transform="matrix(0.99975,0,0,0.99975,0.918515,0.859525)">
```

The browser points that image to the QR Code on the rendered part of the Browser. So I just erased the tag and Voila, the alert image was gone

Now you just have to scan the QR, you can do it with an online tool or just by using your phone and you will get the flag!

Hope you enjoyed the Writeup!