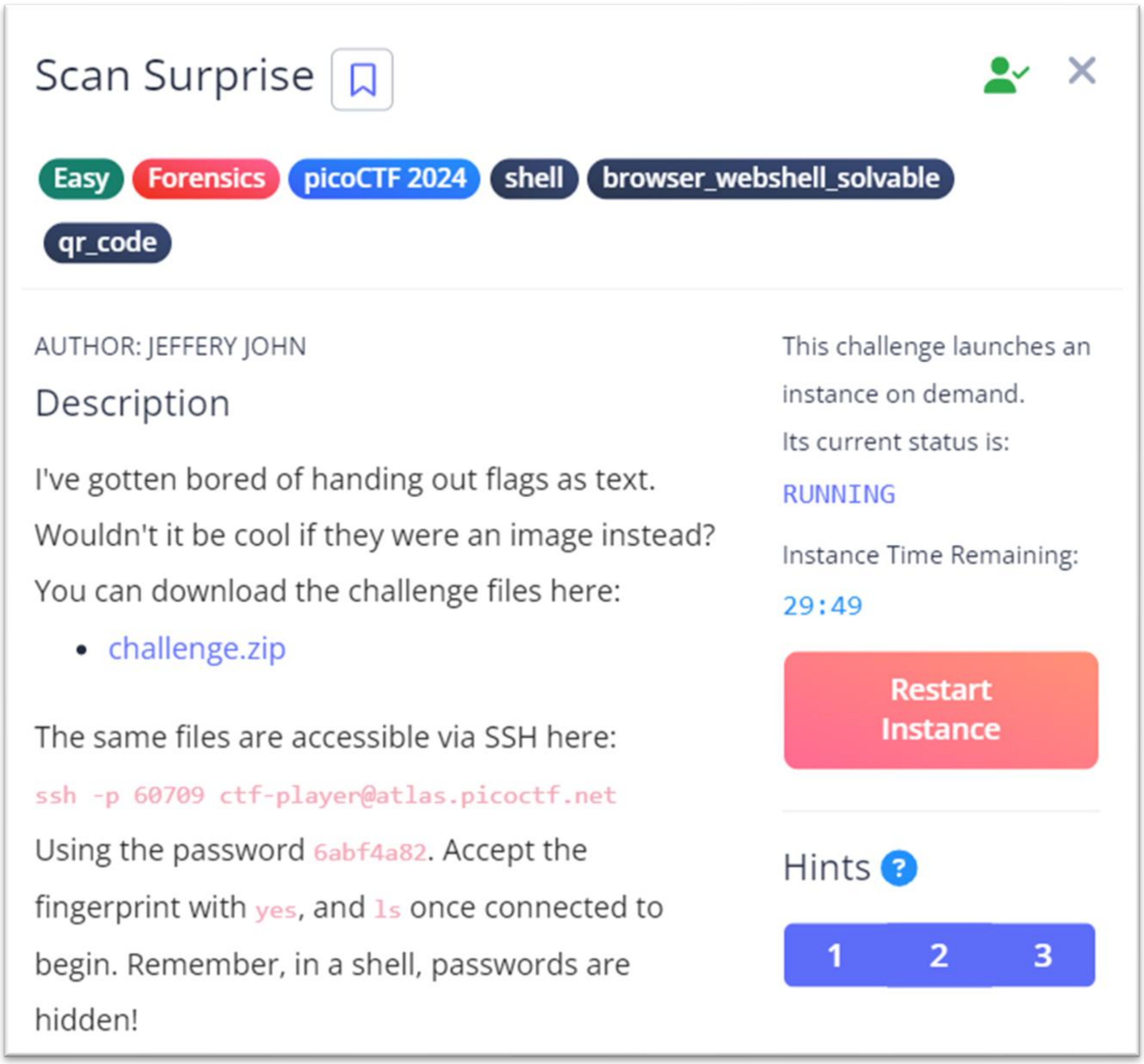


Scan Surprise

AUTHOR: JEFFERY JOHN

Description

I've gotten bored of handing out flags as text. Wouldn't it be cool if they were an image instead? You can download the challenge files here:

- challenge.zip

The same files are accessible via SSH here:

```
ssh -p 60709 ctf-player@atlas.picoctf.net
```

Using the password `6abf4a82`. Accept the fingerprint with `yes`, and `ls` once connected to begin. Remember, in a shell, passwords are hidden!

This challenge launches an instance on demand.
Its current status is:
RUNNING
Instance Time Remaining:
29:49

PICO GYM

Name: Scan Surprise (easy)

Downloadable Files:

[No need to download other files]

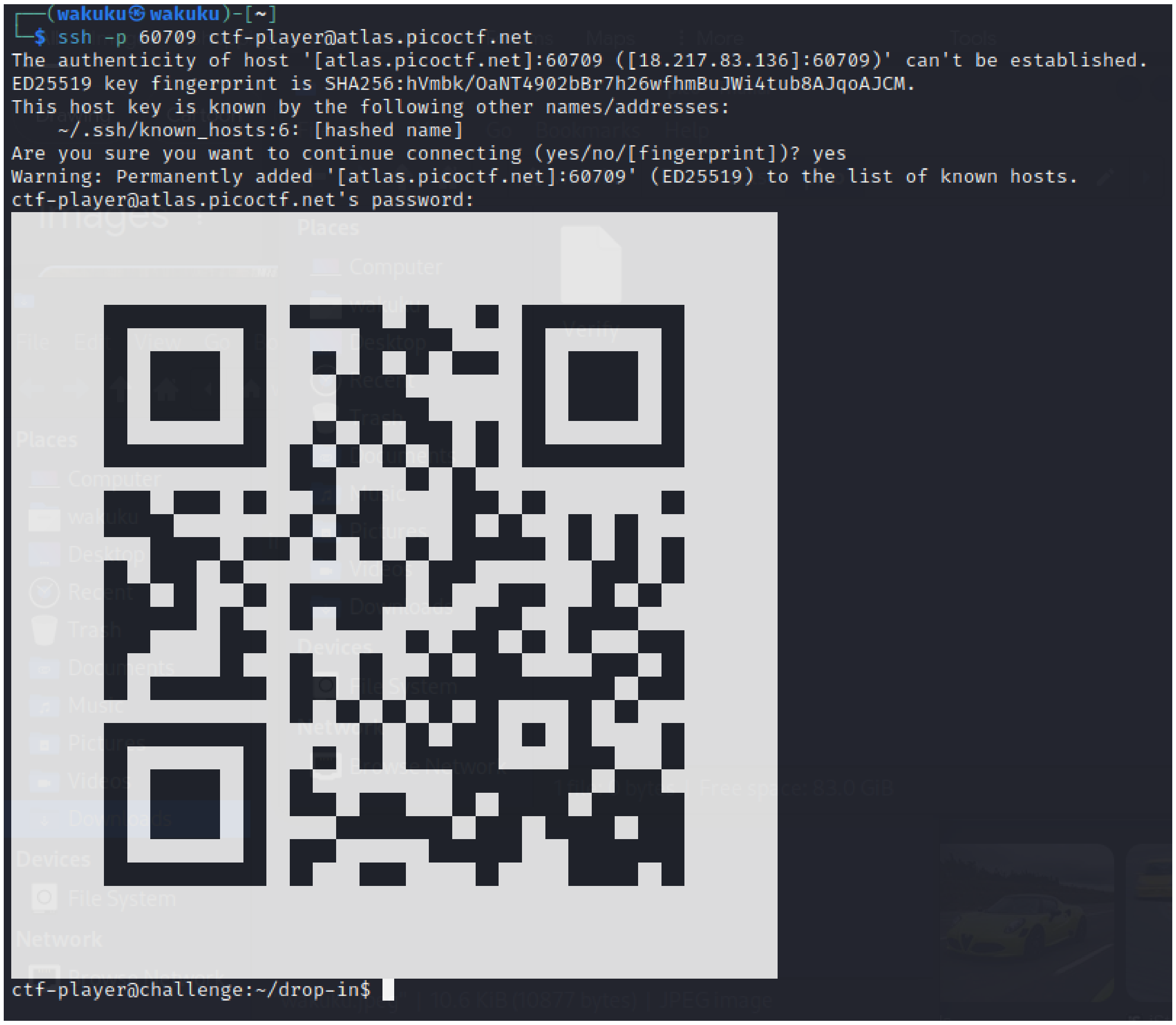
Solution

> After Launching the Instance, I used the ssh command to connect to it.

$ ssh -p 60709 ctf-player@atlas.picoctf.net

Password: 6abf4a82

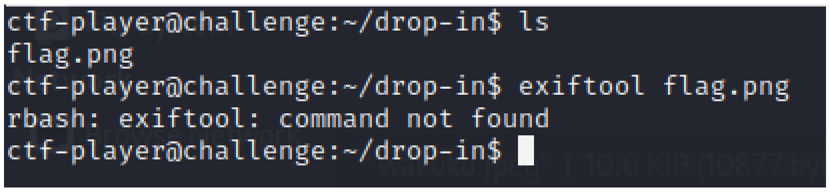
> Once I connected, a qr code pop up inside the terminal



```
┌──(wakuku㉿wakuku)-[~]
└─$ ssh -p 60709 ctf-player@atlas.picoctf.net
The authenticity of host '[atlas.picoctf.net]:60709 ([18.217.83.136]:60709)' can't be established.
ED25519 key fingerprint is SHA256:hVmbk/OaNT4902bBr7h26wfhmBuJWi4tub8AJqoAJCM.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:6: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[atlas.picoctf.net]:60709' (ED25519) to the list of known hosts.
ctf-player@atlas.picoctf.net's password:

ctf-player@challenge:~/drop-in$
```

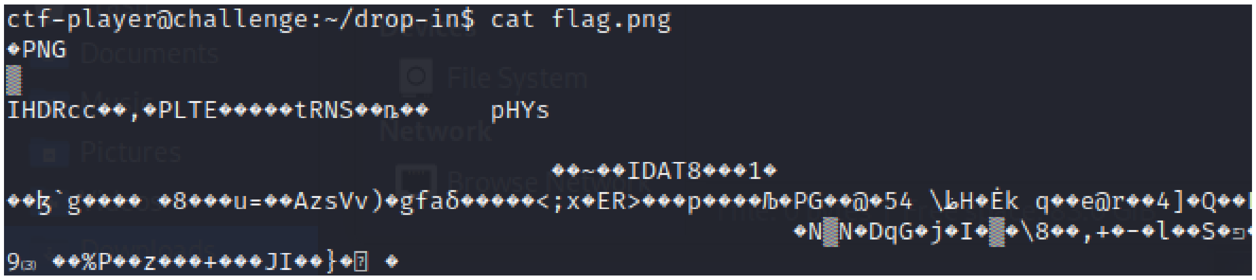
> Now I could use my phone, scar the qr code, and see what it says. But I wanted to see if there are other ways to see whats inside in a forensic way.

> The first thing I did is use the exiftool to see if the flag is hidden in the metadata. However the exiftool is not installed in this instance.



```
ctf-player@challenge:~/drop-in$ ls
flag.png
ctf-player@challenge:~/drop-in$ exiftool flag.png
rbash: exiftool: command not found
ctf-player@challenge:~/drop-in$ █
```

> The next thing I did is see if I could use cat on it. However I only see gibberish



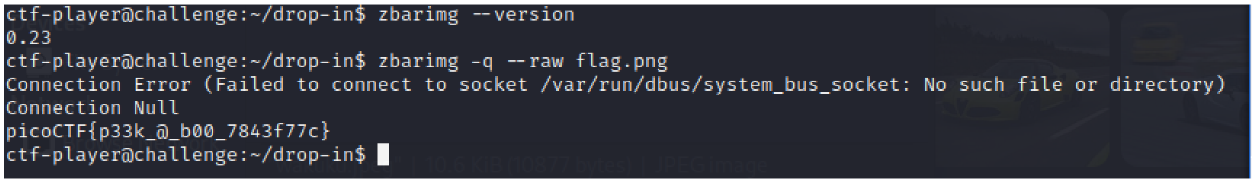
```
ctf-player@challenge:~/drop-in$ cat flag.png
◆PNG
▒

IHDRcc◆◆,◆PLTE◆◆◆◆◆tRNS◆◆ᵰ◆◆    pHYs

                                   ◆◆~◆◆IDAT8◆◆◆1◆
◆◆ɮ`g◆◆◆◆ ◆8◆◆◆u=◆◆AzsVv)◆gfaδ◆◆◆◆◆<;x◆ER>◆◆◆p◆◆◆◆ƀ◆PG◆◆@◆54 \ƚH◆Ėk q◆◆e@r◆◆4]◆Q◆◆
                                                              ◆N▒N◆DqG◆j◆I◆▒◆\8◆◆,+◆-◆l◆◆S◆
9  ◆◆%P◆◆z◆◆◆+◆◆◆JI◆◆}◆  ◆
```

> I then start looking up qr specific tools and here is where I found the command that will solve this problem:

https://medium.com/@sumitdhattarwal4444/creating-and-reading-qr-code-in-linux-5cfeb2d65063

> After verifying that it exists inside the instance, I used this command to get the flag:

```
$ zbarimg -q --raw flag.png
```



```
ctf-player@challenge:~/drop-in$ zbarimg --version
0.23
ctf-player@challenge:~/drop-in$ zbarimg -q --raw flag.png
Connection Error (Failed to connect to socket /var/run/dbus/system_bus_socket: No such file or directory)
Connection Null
picoCTF{p33k_@_b00_7843f77c}
ctf-player@challenge:~/drop-in$ █
```

The Flag is: picoCTF{p33k_@_b00_7843f77c}

Things I’ve learned:

- I learned of a new tool called zbarimg or zbar to read the qr code.
- I also learned qrencode to make a qrcode of a text
- I also learned that when creating a qrcode, you could have its name as a .txt instead of .png or .jpeg